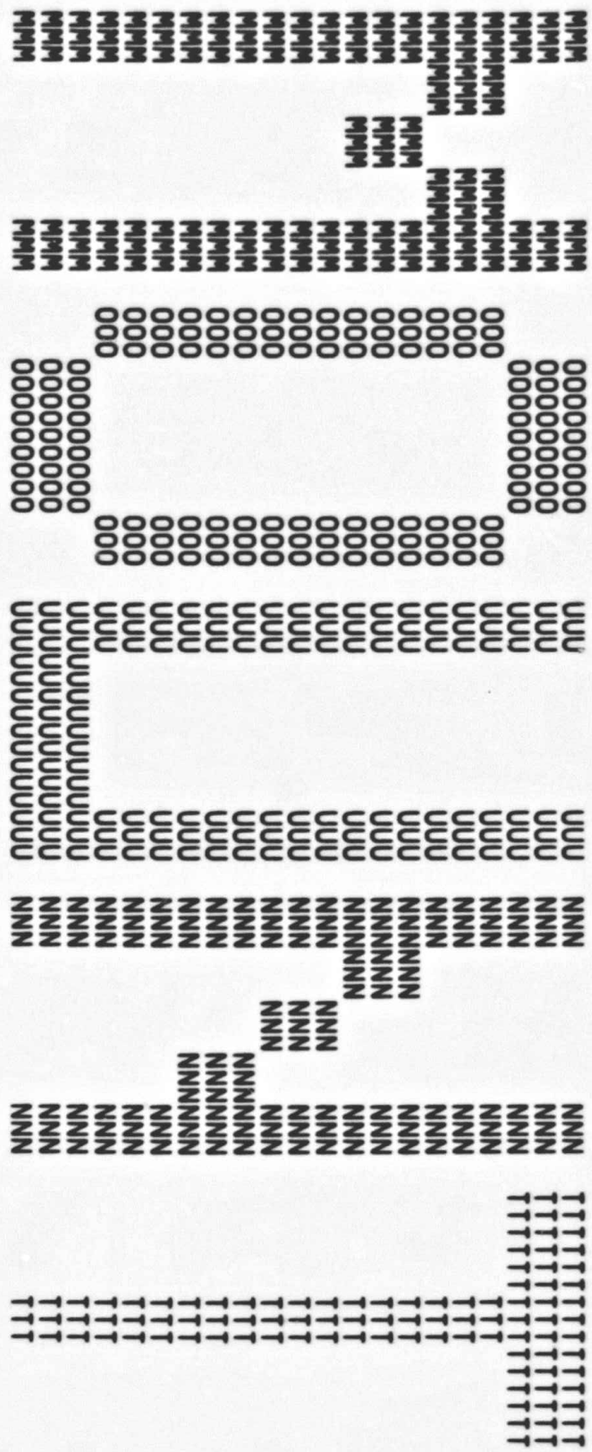

FILEID**CLUSTRMNT

```
  CCCCCCCC  LL          UU      UU    SSSSSSSS  TTTTTTTTTT  RRRRRRRR    MM      MM  NN      NN  TTTTTTTTTT
  CCCCCCCC  LL          UU      UU    SSSSSSSS  TTTTTTTTTT  RRRRRRRR    MM      MM  NN      NN  TTTTTTTTTT
CC          LL          UU      UU  SS              TT      RR      RR  MMMM  MMMM  NN      NN      TT
CC          LL          UU      UU  SS              TT      RR      RR  MMMM  MMMM  NN      NN      TT
CC          LL          UU      UU  SS              TT      RR      RR  MM  MM  MM  NNNN    NN      TT
CC          LL          UU      UU  SS              TT      RR      RR  MM  MM  MM  NNNN    NN      TT
CC          LL          UU      UU    SSSSSS        TT      RRRRRRRR    MM      MM  NN  NN  NN      TT
CC          LL          UU      UU    SSSSSS        TT      RRRRRRRR    MM      MM  NN  NN  NN      TT
CC          LL          UU      UU          SS      TT      RR  RR      MM      MM  NN    NNNN      TT
CC          LL          UU      UU          SS      TT      RR  RR      MM      MM  NN    NNNN      TT
CC          LL          UU      UU          SS      TT      RR    RR    MM      MM  NN      NN      TT
CC          LL          UU      UU          SS      TT      RR    RR    MM      MM  NN      NN      TT
  CCCCCCCC  LLLLLLLLLL  UUUUUUUUUU  SSSSSSSS        TT      RR      RR  MM      MM  NN      NN      TT
  CCCCCCCC  LLLLLLLLLL  UUUUUUUUUU  SSSSSSSS        TT      RR      RR  MM      MM  NN      NN      TT

LL              IIIIII      SSSSSSSS
LL              IIIIII      SSSSSSSS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II        SSSSSS
LL                II        SSSSSS
LL                II              SS
LL                II              SS
LL                II              SS
LL                II              SS
LLLLLLLLLL      IIIIII    SSSSSSSS
LLLLLLLLLL      IIIIII    SSSSSSSS
```

```
;     1          0001  0 MODULE CLUSTRMNT (
;     2          0002  0                 LANGUAGE (BLISS32),
;     3          0003  0                 IDENT = 'V04-001'
;     4          0004  0                 ) =
;     5          0005  1 BEGIN
;     6          0006  1
;     7          0007  1 !
;     8          0008  1 !*****************************************************************************
;     9          0009  1 !*                                                                           *
;    10          0010  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;    11          0011  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;    12          0012  1 !*  ALL RIGHTS RESERVED.                                                      *
;    13          0013  1 !*                                                                           *
;    14          0014  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
;    15          0015  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
;    16          0016  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
;    17          0017  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
;    18          0018  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
;    19          0019  1 !*  TRANSFERRED.                                                              *
;    20          0020  1 !*                                                                           *
;    21          0021  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
;    22          0022  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT     *
;    23          0023  1 !*  CORPORATION.                                                              *
;    24          0024  1 !*                                                                           *
;    25          0025  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
;    26          0026  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
;    27          0027  1 !*                                                                           *
;    28          0028  1 !*                                                                           *
;    29          0029  1 !*****************************************************************************
;    30          0030  1
;    31          0031  1 !++
;    32          0032  1 !
;    33          0033  1 ! FACILITY:  MOUNT Utility Structure Levels 1 & 2
;    34          0034  1 !
;    35          0035  1 ! ABSTRACT:
;    36          0036  1 !
;    37          0037  1 !       This module contains routines used to verify mount consistency
;    38          0038  1 !       throughout a cluster.
;    39          0039  1 !
;    40          0040  1 !
;    41          0041  1 ! ENVIRONMENT:
;    42          0042  1 !
;    43          0043  1 !       STARLET operating system, including privileged system services
;    44          0044  1 !       and internal exec routines.
;    45          0045  1 !
;    46          0046  1 !--
;    47          0047  1 !
;    48          0048  1 !
;    49          0049  1 ! AUTHOR:  Christian D. Saether  CREATION DATE:  5-Aug-1983
;    50          0050  1 !
;    51          0051  1 ! MODIFIED BY:
;    52          0052  1 !
;    53          0053  1 !       V04-001 HH0058          Hai Huang                   13-Sep-1984
;    54          0054  1 !               Do not demote the device lock to CR mode in error path.
;    55          0055  1 !
;    56          0056  1 !       V03-010 HH0054          Hai Huang                   30-Aug-1984
;    57          0057  1 !               Add another sanity check (count the number of device
```

```
;    58          0058  1 !                locks) before making us the first mounter.
;    59          0059  1 !
;    60          0060  1 !       V03-009 HH0053          Hai Huang               29-Aug-1984
;    61          0061  1 !               Clear the device context and make us the first mounter
;    62          0062  1 !               if the device lock value block and the volume lock value
;    63          0063  1 !               block are inconsistent.
;    64          0064  1 !
;    65          0065  1 !       V03-008 HH0045          Hai Huang               10-Aug-1984
;    66          0066  1 !               Take out the volume lock for shared foreign mounts.
;    67          0067  1 !
;    68          0068  1 !       V03-007 HH0042          Hai Huang               27-Jul-1984
;    69          0069  1 !               Define variables so GLOBAL storage can be cleared during
;    70          0070  1 !               run time.
;    71          0071  1 !
;    72          0072  1 !       V03-006 HH0041          Hai Huang               24-Jul-1984
;    73          0073  1 !               Remove REQUIRE 'LIBD$:[VMSLIB.OBJ]MOUNTMSG.B32'.
;    74          0074  1 !
;    75          0075  1 !       V03-005 CDS0003         Christian D. Saether    11-Jul-1984
;    76          0076  1 !               Restore zealous checks on /WRITE, but remove
;    77          0077  1 !               them for /QUOTA until the day the DISKQUOTA utility
;    78          0078  1 !               plays correctly with it.
;    79          0079  1 !
;    80          0080  1 !       V03-004 HH0026          Hai Huang               25-Jun-1984
;    81          0081  1 !               Remove overzealous consistency check on the /[NO]WRITE
;    82          0082  1 !               option.
;    83          0083  1 !
;    84          0084  1 !       V03-003 HH0015          Hai Huang               20-Apr-1984
;    85          0085  1 !               Fix various problems caused by getting generic mount
;    86          0086  1 !               to work.
;    87          0087  1 !
;    88          0088  1 !       V03-002 CDS0002         Christian D. Saether     1-Feb-1984
;    89          0089  1 !               Modify interlock checks to allow multiple writers
;    90          0090  1 !               if mounted /foreign.
;    91          0091  1 !               Add routine headers (comments).
;    92          0092  1 !
;    93          0093  1 !       V03-001 CDS0001         Christian D. Saether     6-Dec-1983
;    94          0094  1 !               Set VCB$V_NOSHARE flag to signal lock are taken
;    95          0095  1 !               in non-shared namespace.
;    96          0096  1 !
;    97          0097  1 !**
;    98          0098  1
;    99          0099  1
;   100          0100  1 LIBRARY 'SYS$LIBRARY:LIB.L32';
;   101          0101  1 REQUIRE 'SRC$:MOUDEF.B32';
;   102          0633  1
;   103          0634  1 OWN
;   104          0635  1         LCKCNT_ITM      : BBLOCK [12 + 4] INITIAL (
;   105          0636  1                           WORD (4),
;   106          0637  1                           WORD (LKI$_LCKCOUNT),
;   107          0638  1                           LONG (0),
;   108          0639  1                           LONG (0),
;   109          0640  1                           LONG (0));
;   110          0641  1
;   111          0642  1 !
;   112          0643  1 ! Note: The following global storage area for various locks is cleared by
;   113          0644  1 ! VMOUNT during run time.
;   114          0645  1 !
```

```
;  115     0646  1 GLOBAL
;  116     0647  1         LCK_GLOBAL_START: VECTOR [0],           ! Mark start of global storage.
;  117     0648  1         DEVLCK_UCB      : REF BBLOCK,           ! UCB of device lock.
;  118     0649  1         DEVLCK_STS      : VECTOR [2, WORD],     ! This MUST precede DEVLCK_LKID.
;  119     0650  1         DEVLCK_LKID,
;  120     0651  1         DEV_CTX         : BBLOCK [16] FIELD (DC),
;  121     0652  1                                                 ! This MUST follow DEVLCK_LKID.
;  122     0653  1         VOLOCK_STS      : VECTOR [2, WORD],     ! This MUST precede VOLOCK_ID.
;  123     0654  1         VOLOCK_ID,                              ! This MUST follow VOLOCK_STS.
;  124     0655  1         VOL_CTX         : BBLOCK [16] FIELD (VC),! This MUST follow VOLOCK_ID.
;  125     0656  1
;  126     0657  1         VOLOCK_COUNT,                           ! Count of volume locks.
;  127     0658  1
;  128     0659  1         VLSETLCK_STS    : VECTOR [2, WORD],     ! This MUST precede VLSETLCK_ID.
;  129     0660  1         VLSETLCK_ID,                            ! This MUST follow VLSETLCK_STS.
;  130     0661  1         VLSETLCK_CTX    : BBLOCK [16] FIELD (VC), ! MUST follow VLSETLCK_ID.
;  131     0662  1         LCK_GLOBAL_END  : VECTOR [0];           ! Mark end of global storage.
;  132     0663  1
```

```
;  134     0664  1 GLOBAL ROUTINE GET_DEVICE_CONTEXT =
;  135     0665  1
;  136     0666  1 !++
;  137     0667  1 !
;  138     0668  1 ! Functional description:
;  139     0669  1 !
;  140     0670  1 ! This routine initializes mount context relevant to the device and
;  141     0671  1 ! volume locks.  It then acquires the device lock value block, if
;  142     0672  1 ! it exists, which contains mount context for that device, if it is
;  143     0673  1 ! mounted already.
;  144     0674  1 !
;  145     0675  1 ! This also interlocks the MOUNT service with the final dismounting
;  146     0676  1 ! functions performed by the file system.
;  147     0677  1 !
;  148     0678  1 ! This routine must be called in kernel mode.
;  149     0679  1 !
;  150     0680  1 ! Calling sequence:
;  151     0681  1 !
;  152     0682  1 !       GET_DEVICE_CONTEXT ()
;  153     0683  1 !
;  154     0684  1 ! Input parameters:
;  155     0685  1 !
;  156     0686  1 !       NONE
;  157     0687  1 !
;  158     0688  1 ! Implicit inputs:
;  159     0689  1 !
;  160     0690  1 !       CHANNEL - Channel on which volume is being mounted.
;  161     0691  1 !
;  162     0692  1 ! Implicit outputs:
;  163     0693  1 !
;  164     0694  1 !       DEV_CTX [DC_FLAGS] - set to zero if first mounter, else contains
;  165     0695  1 !                            the value of pre-existing mounts.
;  166     0696  1 !       VOLOCK_ID       - zeroed
;  167     0697  1 !       VLSETLCK_ID     - zeroed
;  168     0698  1 !       DEVLCK_LKID     - zero if no device lock, else lockid of device lock
;  169     0699  1 !       DEVLCK_UCB      - address of UCB of input CHANNEL
;  170     0700  1 !
;  171     0701  1 ! Routine value:
;  172     0702  1 !
;  173     0703  1 !       Success if no device lock, or if device allocated.
;  174     0704  1 !       Else status of $ENQW, with SS$_VALNOTVALID converted to success.
;  175     0705  1 !
;  176     0706  1 ! Side effects:
;  177     0707  1 !
;  178     0708  1 !       A system owned shared mode device lock (LCK$K_CRMODE) will be
;  179     0709  1 !       converted to a process owned LCK$K_PWMODE lock.  This must
;  180     0710  1 !       be converted back before the MOUNT service completes.
;  181     0711  1 !
;  182     0712  1 !--
;  183     0713  1
;  184     0714  2 BEGIN
;  185     0715  2
;  186     0716  2 LOCAL
;  187     0717  2         STATUS,
;  188     0718  2         STSBLK          : VECTOR [4, WORD];
;  189     0719  2
;  190     0720  2 EXTERNAL
```

```
;  19          0721  2          CHANNEL;
;  192         0722  2
;  193         0723  2 EXTERNAL ROUTINE
;  194         0724  2          GET_CHANNELUCB;
;  195         0725  2
;  196         0726  2 !
;  197         0727  2 ! Mount now directly calls the IOC$SEARCH routine, which returns the
;  198         0728  2 ! lock value block of the device lock. Thus the device lock context
;  199         0729  2 ! should not be unconditionally cleared.
;  200         0730  2 !
;  201         0731  2 ! DEV_CTX [DC_FLAGS] = 0;
;  202         0732  2 !
;  203         0733  2
;  204         0734  2 VOLOCK_ID = 0;
;  205         0735  2 VLSETLCK_ID = 0;
;  206         0736  2 DEVLCK_LKID = 0;
;  207         0737  2
;  208         0738  2 DEVLCK_UCB = GET_CHANNELUCB (.CHANNEL);
;  209         0739  2
;  210         0740  2 IF (DEVLCK_LKID = .DEVLCK_UCB [UCB$L_LOCKID]) EQL 0
;  211         0741  2 THEN
;  212         0742  2     RETURN 1;
;  213         0743  2
;  214         0744  2 ! If the PID field in the ucb is non-zero, then the device is allocated
;  215         0745  2 ! to this process, therefore this is by definition the first mounter
;  216         0746  2 ! on this device.  Because the lock is already held in EX mode, we
;  217         0747  2 ! simply return now and it will be written later.
;  218         0748  2 !
;  219         0749  2
;  220         0750  2 IF .DEVLCK_UCB [UCB$L_PID] NEQ 0
;  221         0751  2 THEN
;  222         0752  2     RETURN 1;
;  223         0753  2
;  224         0754  2 ! Get the device lock in PW mode.  This both gets the current contents
;  225         0755  2 ! of the value block, and gets it in a mode from which it can be written
;  226         0756  2 ! later.
;  227         0757  2 ! This is also necessary to interlock with the file system completing
;  228         0758  2 ! the last dismount on a device.  In that case, the CHECK_DISMOUNT
;  229         0759  2 ! routine in the file system will want to clear the value block to
;  230         0760  2 ! remove the mount context information.  It must do this because the
;  231         0761  2 ! device lock itself does not disappear until the last channel is
;  232         0762  2 ! deassigned, and the mount context in the device lock value block
;  233         0763  2 ! must be cleared when the last dismount occurs.
;  234         0764  2 !
;  235         0765  2
;  236       P 0766  2 STATUS = $ENQW (LKMODE = LCK$K_PWMODE,
;  237       P 0767  2                 LKSB = DEVLCK_STS,
;  238       P 0768  2                 EFN = MOUNT_EFN,
;  239       P 0769  2                 FLAGS = LCK$M_CONVERT + LCK$M_SYNCSTS + LCK$M_VALBLK
;  240         0770  2                       + LCK$M_NOQUOTA);
;  241         0771  2
;  242         0772  2 IF NOT .STATUS
;  243         0773  2 THEN
;  244         0774  2     RETURN .STATUS;
;  245         0775  2
;  246         0776  3 IF (STATUS = .DEVLCK_STS [0])
;  247         0777  2 THEN
```

```
;  248          0778  2      RETURN .STATUS;
;  249          0779  2
;  250          0780  2  IF .STATUS<0,16> EQL SS$_VALNOTVALID
;  251          0781  2  THEN
;  252          0782  2      STATUS = 1;
;  253          0783  2
;  254          0784  2  .STATUS
;  255          0785  2
;  256          0786  1  END;                 ! of routine GET_DEVICE_CONTEXT

                                                        .TITLE  CLUSTRMNT
                                                        .IDENT  \V04-001\

                                                        .PSECT  $OWN$,NOEXE,2

                                            0004  00000 LCKCNT_ITM:
                                                        .WORD   4                       ;
                                              0205  00002       .WORD   517             ;
                                        00000000  00004       .LONG   0               ;
                                        00000000  00008       .LONG   0               ;
                                        00000000  0000C       .LONG   0               ;

                                                        .PSECT  $GLOBAL$,NOEXE,2

                                                  00000 LCK_GLOBAL_START::
                                                        .BLKB   0
                                                  00000 DEVLCK_UCB::
                                                        .BLKB   4
                                                  00004 DEVLCK_STS::
                                                        .BLKB   4
                                                  00008 DEVLCK_LKID::
                                                        .BLKB   4
                                                  0000C DEV_CTX::
                                                        .BLKB   16
                                                  0001C VOLOCK_STS::
                                                        .BLKB   4
                                                  00020 VOLOCK_ID::
                                                        .BLKB   4
                                                  00024 VOL_CTX::
                                                        .BLKB   16
                                                  00034 VOLOCK_COUNT::
                                                        .BLKB   4
                                                  00038 VLSETLCK_STS::
                                                        .BLKB   4
                                                  0003C VLSETLCK_ID::
                                                        .BLKB   4
                                                  00040 VLSETLCK_CTX::
                                                        .BLKB   16
                                                  00050 LCK_GLOBAL_END::
                                                        .BLKB   0

                                                        .EXTRN  CHANNEL, GET_CHANNELUCB
                                                        .EXTRN  SYS$ENQW

                                                        .PSECT  $CODE$,NOWRT,2
```

```
                                 0004 00000          .ENTRY  GET_DEVICE_CONTEXT, Save R2                     ; 0664
                          52     0000' CF   9E 00002         MOVAB   DEVLCK_LKID, R2                         ;
                          5E           08   C2 00007         SUBL2   #8, SP                                  ;
                                    18   A2   D4 0000A         CLRL    VOLOCK_ID                               ; 0734
                                    34   A2   D4 0000D         CLRL    VLSETLCK_ID                             ; 0735
                                    62   D4   D4 00010         CLRL    DEVLCK_LKID                             ; 0736
                                 0000G   CF   DD 00012         PUSHL   CHANNEL                                 ; 0738
                               0000G  CF      01   FB 00016         CALLS   #1, GET_CHANNELUCB                      ;
                                 F8   A2      50   D0 0001B         MOVL    R0, DEVLCK_UCB                          ;
                                 62        20   A0   D0 0001F         MOVL    32(R0), DEVLCK_LKID                     ; 0740
                                           2D   13   00023         BEQL    1$                                      ;
                                    2C   A0   D5   00025         TSTL    44(R0)                                  ; 0750
                                           28   12   00028         BNEQ    1$                                      ;
                                           7E   7C   0002A         CLRQ    -(SP)                                   ; 0770
                                           7E   7C   0002C         CLRQ    -(SP)                                   ;
                                           7E   7C   0002E         CLRQ    -(SP)                                   ;
                                    7E           2B   7D   00030         MOVQ    #43, -(SP)                      ;
                                       FC   A2   9F   00033         PUSHAB  DEVLCK_STS                              ;
                                           04   DD   00036         PUSHL   #4                                      ;
                                           1A   DD   00038         PUSHL   #26                                     ;
                          00000000G        00   0B   FB 0003A         CALLS   #11, SYS$ENQW                           ;
                                           11   50   E9 00041         BLBC    STATUS, 2$                              ; 0772
                                           50   FC   A2   3C 00044         MOVZWL  DEVLCK_STS, STATUS              ; 0776
                                           0A   50   E8 00048         BLBS    STATUS, 2$                              ;
                                 09F0      8F   50   B1 0004B         CMPW    STATUS, #2544                           ; 0780
                                           03   12   00050         BNEQ    2$                                      ;
                                    50           01   D0 00052 1$:      MOVL    #1, STATUS                      ; 0782
                                                 04   00055 2$:      RET                                     ; 0786

; Routine Size:  86 bytes,    Routine Base:  $CODE$ + 0000

;  257                0787  1
```

```
;  259          0788  1 GLOBAL ROUTINE CHECK_CLUSTER_SANITY : NOVALUE =
;  260          0789  1
;  261          0790  1 !++
;  262          0791  1 !
;  263          0792  1 ! Functional description:
;  264          0793  1 !
;  265          0794  1 ! This routine enforces consistency between the current mount
;  266          0795  1 ! request and mounts that have already been executed for this
;  267          0796  1 ! device on other nodes in the cluster.  It does so by comparing
;  268          0797  1 ! information from this request with the value block of the
;  269          0798  1 ! device lock (DEV_CTX) and signalling the appropriate error
;  270          0799  1 ! if they are inconsistent.
;  271          0800  1 !
;  272          0801  1 ! Input parameters:
;  273          0802  1 !       NONE
;  274          0803  1 !
;  275          0804  1 ! Implicit inputs:
;  276          0805  1 !
;  277          0806  1 !       MOUNT_OPTIONS - bitvector
;  278          0807  1 !               OPT_FOREIGN
;  279          0808  1 !               OPT_WRITE
;  280          0809  1 !               OPT_GROUP
;  281          0810  1 !               OPT_SYSTEM
;  282          0811  1 !               OPT_NOQUOTA
;  283          0812  1 !               OPT_PROTECTION
;  284          0813  1 !               OPT_OWNER_UIC
;  285          0814  1 !       DEV_CTX - device lock value block
;  286          0815  1 !               DC_FOREIGN
;  287          0816  1 !               DC_NOINTERLOCK
;  288          0817  1 !               DC_GROUP
;  289          0818  1 !               DC_SYSTEM
;  290          0819  1 !               DC_WRITE
;  291          0820  1 !               DC_NOQUOTA
;  292          0821  1 !               DC_OVR_PROT
;  293          0822  1 !               DC_PROTECTION
;  294          0823  1 !               DC_OVR_OWNUIC
;  295          0824  1 !               DC_OWNER_UIC
;  296          0825  1 !       PROTECTION - desired protection mask for volume
;  297          0826  1 !       OWNER_UIC - owner UIC of volume
;  298          0827  1 !       STORED_CONTEXT - bitvector
;  299          0828  1 !               XQP - this is an XQP (as opposed to ACP)
;  300          0829  1 !
;  301          0830  1 ! Output parameters:
;  302          0831  1 !       NONE
;  303          0832  1 !
;  304          0833  1 ! Routine value:
;  305          0834  1 !       NONE
;  306          0835  1 !
;  307          0836  1 ! Side effects:
;  308          0837  1 !       Signals an error condition if parameters inconsistent with
;  309          0838  1 !       pre-existing mount of this device on another node.
;  310          0839  1 !
;  311          0840  1 !--
;  312          0841  1
;  313          0842  2 BEGIN
;  314          0843  2
;  315          0844  2 EXTERNAL
```

```
;  316          0845  2          MOUNT_OPTIONS   : BITVECTOR,
;  317          0846  2          STORED_CONTEXT  : BITVECTOR,
;  318          0847  2          PROTECTION      : WORD,
;  319          0848  2          OWNER_UIC;
;  320          0849  2
;  321          0850  2 LOCAL
;  322          0851  2          STATUS,
;  323          0852  2          DESC    : INITIAL (0);
;  324          0853  2
;  325          0854  2 LABEL
;  326          0855  2          TESTS;
;  327          0856  2
;  328          0857  2 TESTS:
;  329          0858  3 BEGIN
;  330          0859  3
;  331          0860  3 ! Everyone mounts /foreign or no one mounts /foreign.
;  332          0861  3 !
;  333          0862  3
;  334          0863  3 IF .DEV_CTX [DC_FOREIGN] NEQ .MOUNT_OPTIONS [OPT_FOREIGN]
;  335          0864  3 THEN
;  336          0865  4     BEGIN
;  337          0866  5     DESC = (IF .DEV_CTX [DC_FOREIGN]
;  338          0867  5                THEN DESCRIPTOR ('/FOREIGN')
;  339          0868  4                ELSE DESCRIPTOR ('/NOFOREIGN'));
;  340          0869  4     STATUS = MOUN$_INCONFOR;
;  341          0870  4     LEAVE TESTS
;  342          0871  3     END;
;  343          0872  3
;  344          0873  3 ! NOINTERLOCK means it is not mounted with an xqp, and hence
;  345          0874  3 ! does not synchronize access to the volume.  If an ACP is used,
;  346          0875  3 ! only one writer is allowed.  If mounted foreign, multiple writers
;  347          0876  3 ! are allowed (you're on your own).  If mounted with the xqp anywhere,
;  348          0877  3 ! (not NOINTERLOCK), it must be mounted with the xqp everywhere (this
;  349          0878  3 ! is only possible with the /proc=f11bacp switch, and f11bacp is
;  350          0879  3 ! already gone as of field test 1).
;  351          0880  3 !
;  352          0881  3
;  353          0882  4 IF (.DEV_CTX [DC_NOINTERLOCK]
;  354          0883  5          AND (.MOUNT_OPTIONS [OPT_WRITE] AND .DEV_CTX [DC_WRITE])
;  355          0884  4          AND NOT .MOUNT_OPTIONS [OPT_FOREIGN])
;  356          0885  4     OR (NOT .DEV_CTX [DC_NOINTERLOCK] AND NOT .STORED_CONTEXT [XQP])
;  357          0886  3 THEN
;  358          0887  4     BEGIN
;  359          0888  4     STATUS = MOUN$_INCOMPACP;
;  360          0889  4     LEAVE TESTS
;  361          0890  3     END;
;  362          0891  3
;  363          0892  3 ! If this is not an xqp, it is an ods-1 volume, and the remaining
;  364          0893  3 ! checks are not relevant.
;  365          0894  3 !
;  366          0895  3
;  367          0896  3 IF NOT .STORED_CONTEXT [XQP]
;  368          0897  3 THEN
;  369          0898  3     RETURN;
;  370          0899  3
;  371          0900  3 IF .DEV_CTX [DC_GROUP] NEQ .MOUNT_OPTIONS [OPT_GROUP]
;  372          0901  3          OR .DEV_CTX [DC_SYSTEM] NEQ .MOUNT_OPTIONS [OPT_SYSTEM]
```

```
;  373          0902  3 THEN
;  374          0903  4     BEGIN
;  375          0904  5     DESC = (IF .DEV_CTX [DC_GROUP]
;  376          0905  5                 THEN DESCRIPTOR ('/GROUP')
;  377          0906  5                 ELSE IF .DEV_CTX [DC_SYSTEM]
;  378          0907  5                 THEN DESCRIPTOR ('/SYSTEM')
;  379          0908  4                 ELSE DESCRIPTOR ('/SHARE'));
;  380          0909  4     STATUS = MOUN$_INCONSHR;
;  381          0910  4     LEAVE TESTS
;  382          0911  3     END;
;  383          0912  3
;  384          0913  3 !
;  385          0914  3 ! Ironically, the following consistency check caused mount to be
;  386          0915  3 ! inconsistent with respect treatment of a physically write-locked disk.
;  387          0916  3 ! When mounting a write-locked disk cluster-wide without the /NOWRITE
;  388          0917  3 ! qualifier, the first node to attempt the mount succeeds with a warning
;  389          0918  3 ! that the device is write-locked. Subsequent nodes will fail with the
;  390          0919  3 ! "Inconsistent /WRITE option, cluster mounted /NOWRITE" error.  For this
;  391          0920  3 ! reason, we remove this overzealous consistency check.
;  392          0921  3 !
;  393          0922  3 ! The call to this routine in MOUNT_DISK2 has been moved to beyond the
;  394          0923  3 ! point where we have determined whether the disk is physically writeable
;  395          0924  3 ! or not.  This eliminates the problem discussed above, so the check
;  396          0925  3 ! goes back in until we can figure out why it makes sense to allow
;  397          0926  3 ! a mix of /write and /nowrite.
;  398          0927  3
;  399          0928  3 IF .DEV_CTX [DC_WRITE] NEQ .MOUNT_OPTIONS [OPT_WRITE]
;  400          0929  3 THEN
;  401          0930  4     BEGIN
;  402          0931  5     DESC = (IF .DEV_CTX [DC_WRITE]
;  403          0932  5                 THEN DESCRIPTOR ('/WRITE')
;  404          0933  4                 ELSE DESCRIPTOR ('/NOWRITE'));
;  405          0934  4     STATUS = MOUN$_INCONWRITE;
;  406          0935  4     LEAVE TESTS
;  407          0936  3     END;
;  408          0937  3
;  409          0938  3 ! As of field test 1, this check is incomplete in that the
;  410          0939  3 ! DISKQUOTA utility can modify whether quotas are enabled or
;  411          0940  3 ! not, and does not respect or modify this device lock value block flag.
;  412          0941  3 !
;  413          0942  3 ! So lets take the check out until we know how to do it right.
;  414          0943  3 !
;  415          0944  3
;  416          0945  3 !IF .DEV_CTX [DC_NOQUOTA] NEQ .MOUNT_OPTIONS [OPT_NOQUOTA]
;  417          0946  3 !THEN
;  418          0947  3 !    BEGIN
;  419          0948  3 !    DESC = (IF .DEV_CTX [DC_NOQUOTA]
;  420          0949  3 !                THEN DESCRIPTOR ('/NOQUOTA')
;  421          0950  3 !                ELSE DESCRIPTOR ('/QUOTA'));
;  422          0951  3 !    STATUS = MOUN$_INCONQUOTA;
;  423          0952  3 !    LEAVE TESTS
;  424          0953  3 !    END;
;  425          0954  3
;  426          0955  3 IF .DEV_CTX [DC_OVR_PROT] NEQ .MOUNT_OPTIONS [OPT_PROTECTION]
;  427          0956  4     OR (.DEV_CTX [DC_PROTECTION] NEQ .PROTECTION)
;  428          0957  3 THEN
;  429          0958  4     BEGIN
```

```
;  430          0959  4     STATUS = MOUN$_INCONPROT;
;  431          0960  4     LEAVE TESTS
;  432          0961  3     END;
;  433          0962  3
;  434          0963  3 IF .DEV_CTX [DC_OVR_OWNUIC] NEQ .MOUNT_OPTIONS [OPT_OWNER_UIC]
;  435          0964  4     OR (.DEV_CTX [DC_OWNER_UIC] NEQ .OWNER_UIC)
;  436          0965  3 THEN
;  437          0966  4     BEGIN
;  438          0967  4     STATUS = MOUN$_INCONOWNER;
;  439          0968  4     LEAVE TESTS
;  440          0969  3     END;
;  441          0970  3
;  442          0971  3 ! Passed all the consistency tests.
;  443          0972  3 ! Return.
;  444          0973  3 !
;  445          0974  3
;  446          0975  3 RETURN;
;  447          0976  2 END;                        ! of block TESTS
;  448          0977  2
;  449          0978  2 ! If here, there was a problem.  Signal the error.
;  450          0979  2 !
;  451          0980  2
;  452          0981  2 IF .DESC EQL 0
;  453          0982  2 THEN
;  454          0983  3     BEGIN
;  455          0984  3     ERR_EXIT (.STATUS);
;  456          0985  3     RETURN
;  457          0986  3     END
;  458          0987  2 ELSE
;  459          0988  3     BEGIN
;  460          0989  3     ERR_EXIT (.STATUS, 2, .(.DESC)<0,16>, .(.DESC + 4));
;  461          0990  3     RETURN
;  462          0991  2     END;
;  463          0992  2
;  464          0993  1 END;                        ! of shared mount cluster consistency checks.


                                                    .PSECT  $PLIT$,NOWRT,NOEXE,2

                    4E  47  49  45  52  4F  46  2F  00000 P.AAB:   .ASCII  \/FOREIGN\                ;
                                      00000008      00008 P.AAA:   .LONG   8                        ;
                                      00000000'     0000C         .ADDRESS P.AAB                    ;
                4E  47  49  45  52  4F  46  4F  4E  2F  00010 P.AAD:   .ASCII  \/NOFOREIGN\          ;
                                                    0001A         .BLKB   2
                                      0000000A      0001C P.AAC:   .LONG   10                       ;
                                      00000000'     00020         .ADDRESS P.AAD                    ;
                        50  55  4F  52  47  2F      00024 P.AAF:   .ASCII  \/GROUP\                 ;
                                                    0002A         .BLKB   2
                                      00000006      0002C P.AAE:   .LONG   6                        ;
                                      00000000'     00030         .ADDRESS P.AAF                    ;
                    4D  45  54  53  59  53  2F      00034 P.AAH:   .ASCII  \/SYSTEM\                ;
                                                    0003B         .BLKB   1
                                      00000007      0003C P.AAG:   .LONG   7                        ;
                                      00000000'     00040         .ADDRESS P.AAH                    ;
                        45  52  41  48  53  2F      00044 P.AAJ:   .ASCII  \/SHARE\                 ;
                                                    0004A         .BLKB   2
```

```
                                00000006  0004C P.AAI:  .LONG   6                                                       ;
                                00000000' 00050         .ADDRESS P.AAJ                                                ;
                          45 54 49 52 57 2F 00054 P.AAL:  .ASCII  \/WRITE\                                            ;
                                            0005A         .BLKB   2                                                     ;
                                00000006  0005C P.AAK:  .LONG   6                                                     ;
                                00000000' 00060         .ADDRESS P.AAL                                                ;
                       45 54 49 52 57 4F 4E 2F 00064 P.AAN:  .ASCII  \/NOWRITE\                                       ;
                                00000008  0006C P.AAM:  .LONG   8                                                     ;
                                00000000' 00070         .ADDRESS P.AAN                                                ;

                                                        .EXTRN  MOUNT_OPTIONS, STORED_CONTEXT
                                                        .EXTRN  PROTECTION, OWNER_UIC

                                                        .PSECT  $CODE$,NOWRT,2

                                        00FC 00000      .ENTRY  CHECK_CLUSTER_SANITY, Save R2,R3,R4,R5,R6,-           ; 0788
                                                                R7                                                    ;
                          57 00000000G  00   9E 00002   MOVAB   LIB$STOP, R7                                          ;
                          56     0000'  CF   9E 00009   MOVAB   P.AAA, R6                                            ;
                          55     0000G  CF   9E 0000E   MOVAB   MOUNT_OPTIONS, R5                                    ;
                          54     0000'  CF   9E 00013   MOVAB   DEV_CTX, R4                                          ;
                                        52   D4 00018   CLRL    DESC                                                  ; 0842
                50 01 A5        01      03   EF 0001A   EXTZV   #3, #1, MOUNT_OPTIONS+1, R0                          ; 0863
                50    64        01      01   ED 00020   CMPZV   #1, #1, DEV_CTX, R0                                   ;
                                        16   13 00025   BEQL    3$                                                    ;
                      05        64      01   E1 00027   BBC     #1, DEV_CTX, 1$                                       ; 0866
                                52      66   9E 0002B   MOVAB   P.AAA, DESC                                         ; 0867
                                        04   11 0002E   BRB     2$                                                    ;
                                52      14   A6 9E 00030 1$:   MOVAB   P.AAC, DESC                                  ; 0868
                                53 0072825C  8F   D0 00034 2$:  MOVL    #7504476, STATUS                              ; 0869
                                        64   11 0003B   BRB     13$                                                   ; 0870
                                12      01   A4 E9 0003D 3$:   BLBC    DEV_CTX+1, 5$                                 ; 0882
                   09 01        A5      01   E1 00041   BBC     #1, MOUNT_OPTIONS+1, 4$                               ; 0883
                   05           64      04   E1 00046   BBC     #4, DEV_CTX, 4$                                       ;
                   0A 01        A5      03   E1 0004A   BBC     #3, MOUNT_OPTIONS+1, 6$                               ; 0884
                               0F   01   A4 E8 0004F 4$:   BLBS    DEV_CTX+1, 7$                                 ; 0885
                   10 0000G    CF      02   E0 00053 5$:   BBS     #2, STORED_CONTEXT, 8$                                ;
                                53 007280A4  8F   D0 00059 6$:  MOVL    #7504036, STATUS                              ; 0888
                                        63   11 00060   BRB     17$                                                   ; 0889
                   01 0000G    CF              02   E0 00062 7$:   BBS     #2, STORED_CONTEXT, 8$                    ; 0896
                                                04 00068   RET
                50    65        01              07   EF 00069 8$:   EXTZV   #7, #1, MOUNT_OPTIONS, R0                 ; 0900
                50    64        01              02   ED 0006E   CMPZV   #2, #1, DEV_CTX, R0                          ;
                                                0D   12 00073   BNEQ    9$                                            ;
                50 01 A5        01              00   EF 00075   EXTZV   #0, #1, MOUNT_OPTIONS+1, R0                 ; 0901
                50    64        01              03   ED 0007B   CMPZV   #3, #1, DEV_CTX, R0                          ;
                                                21   13 00080   BEQL    14$                                           ;
                      06        64              02   E1 00082 9$:   BBC     #2, DEV_CTX, 10$                          ; 0904
                                52      24      A6   9E 00086   MOVAB   P.AAE, DESC                                 ; 0905
                                                0E   11 0008A   BRB     12$                                           ;
                      06        64              03   E1 0008C 10$:  BBC     #3, DEV_CTX, 11$                          ; 0906
                                52      34      A6   9E 00090   MOVAB   P.AAG, DESC                                 ; 0907
                                                04   11 00094   BRB     12$                                           ;
                                52      44      A6   9E 00096 11$:  MOVAB   P.AAI, DESC                             ; 0908
                                53 00728234     8F   D0 0009A 12$:  MOVL    #7504436, STATUS                        ; 0909
                                                5E   11 000A1 13$:  BRB     22$                                       ; 0910
                50 01 A5        01              01   EF 000A3 14$:  EXTZV   #1, #1, MOUNT_OPTIONS+1, R0             ; 0928
```

```
                50              64              01              04 ED 000A9          CMPZV   #4, #1, DEV_CTX, R0                     ;
                                                                17 13 000AE          BEQL    18$                                     ;
                                06              64              04 E1 000B0          BBC     #4, DEV_CTX, 15$                        ; 0931
                                                52        54    A6 9E 000B4          MOVAB   P.AAK, DESC                             ; 0932
                                                                04 11 000B8          BRB     16$                                     ;
                                                52        64    A6 9E 000BA 15$:     MOVAB   P.AAM, DESC                             ; 0933
                                                53 00728254     8F D0 000BE 16$:     MOVL    #7504468, STATUS                        ; 0934
                                                                3A 11 000C5 17$:     BRB     22$                                     ; 0935
                50      02      A5              01              01 EF 000C7 18$:     EXTZV   #1, #1, MOUNT_OPTIONS+2, R0             ; 0955
                50              64              01              06 ED 000CD          CMPZV   #6, #1, DEV_CTX, R0                     ;
                                                                08 12 000D2          BNEQ    19$                                     ;
                                        0000G   CF        02    A4 B1 000D4          CMPW    DEV_CTX+2, PROTECTION                   ; 0956
                                                                09 13 000DA          BEQL    20$                                     ;
                                                53 0072823C     8F D0 000DC 19$:     MOVL    #7504444, STATUS                        ; 0959
                                                                1C 11 000E3          BRB     22$                                     ; 0960
                50      02      A5              01              02 EF 000E5 20$:     EXTZV   #2, #1, MOUNT_OPTIONS+2, R0             ; 0963
                50              64              01              07 ED 000EB          CMPZV   #7, #1, DEV_CTX, R0                     ;
                                                                08 12 000F0          BNEQ    21$                                     ;
                                        0000G   CF        04    A4 D1 000F2          CMPL    DEV_CTX+4, OWNER_UIC                    ; 0964
                                                                1E 13 000F8          BEQL    24$                                     ;
                                                53 0072824C     8F D0 000FA 21$:     MOVL    #7504460, STATUS                        ; 0967
                                                                52 D5 00101 22$:     TSTL    DESC                                    ; 0981
                                                                06 12 00103          BNEQ    23$                                     ;
                                                                53 DD 00105          PUSHL   STATUS                                  ; 0984
                                                67              01 FB 00107          CALLS   #1, LIB$STOP                            ;
                                                                04 0010A             RET                                             ; 0988
                                                        04      A2 DD 0010B 23$:     PUSHL   4(DESC)                                 ; 0989
                                                7E              62 3C 0010E          MOVZWL  (DESC), -(SP)                           ;
                                                                02 DD 00111          PUSHL   #2                                      ;
                                                                53 DD 00113          PUSHL   STATUS                                  ;
                                                67              04 FB 00115          CALLS   #4, LIB$STOP                            ;
                                                                04 00118 24$:        RET                                             ; 0993

; Routine Size:  281 bytes,    Routine Base:  $CODE$ + 0056

;  465          0994  1
```

```
;  467     0995  1 GLOBAL ROUTINE STORE_CONTEXT : NOVALUE =
;  468     0996  1
;  469     0997  1 !++
;  470     0998  1 !
;  471     0999  1 ! Functional description:
;  472     1000  1 !
;  473     1001  1 ! This routine stores the various value block contexts by converting
;  474     1002  1 ! the volume, volume set (if present), and device locks to their
;  475     1003  1 ! system owned, compatible modes.  The order is which the locks are
;  476     1004  1 ! released is important because the mount kernel mode handler needs
;  477     1005  1 ! to know how to clean up if anything goes wrong.
;  478     1006  1 !
;  479     1007  1 ! Volume and volume set locks may not be present if this volume
;  480     1008  1 ! is serviced with an acp (ods-1, or /proc=filbacp).  The device
;  481     1009  1 ! lock may not be present if the device is not cluster accessible.
;  482     1010  1 !
;  483     1011  1 ! This routine is called in kernel mode.
;  484     1012  1 !
;  485     1013  1 ! Input parameters:
;  486     1014  1 !       NONE
;  487     1015  1 !
;  488     1016  1 ! Implicit inputs:
;  489     1017  1 !
;  490     1018  1 !       VOLOCK_ID - nonzero if a volume lock is present
;  491     1019  1 !       VOL_CTX - volume context (value block), all of it
;  492     1020  1 !               specifically referenced in this routine -
;  493     1021  1 !               VC_NOTFIRST_MNT - clear if this is the first mounter
;  494     1022  1 !       VLSETLCK_ID - nonzero if a volume set lock is present
;  495     1023  1 !       VLSETLCK_CTX - volume set context (value block), all of it
;  496     1024  1 !               specifically referenced in this routine -
;  497     1025  1 !               VC_NOTFIRST_MNT - clear if this is the first mounter
;  498     1026  1 !       DEVLCK_LKID - nonzero if device lock is present
;  499     1027  1 !       DEV_CTX - device lock value block (mount context)
;  500     1028  1 !               DC_NOTFIRST_MNT - clear if this is the first mounter
;  501     1029  1 !       STORED_CONTEXT - bitvector
;  502     1030  1 !               XQP - set if this is an XQP
;  503     1031  1 !       MOUNT_OPTIONS - bitvector
;  504     1032  1 !               OPT_FOREIGN
;  505     1033  1 !               OPT_WRITE
;  506     1034  1 !               OPT_GROUP
;  507     1035  1 !               OPT_SYSTEM
;  508     1036  1 !               OPT_NOQUOTA
;  509     1037  1 !               OPT_PROTECTION
;  510     1038  1 !               OPT_OWNER_UIC
;  511     1039  1 !       PROTECTION - protection mask applied to the volume
;  512     1040  1 !       OWNER_UIC - owner UIC of the volume
;  513     1041  1 !
;  514     1042  1 ! Output parameters:
;  515     1043  1 !       NONE
;  516     1044  1 !
;  517     1045  1 ! Implicit outputs:
;  518     1046  1 !
;  519     1047  1 !       VOLOCK_ID - zeroed if all locks successfully converted
;  520     1048  1 !       VLSETLCK_ID - zeroed if all locks successfully converted
;  521     1049  1 !       DEVLCK_LKID - zeroed if all locks successfully converted
;  522     1050  1 !       REAL_RVT [RVT$L_STRUCLKID] - lock ID of volume set lock
;  523     1051  1 !       VOL_CTX [VC_NOTFIRST_MNT] - set to 1
```

```
;  524          1052  1 !       DEV_CTX [DC_NOTFIRST_MNT] - set to 1
;  525          1053  1 !       DEV_CTX - following fields are set as appropriate if first mounter
;  526          1054  1 !               DC_FOREIGN
;  527          1055  1 !               DC_WRITE
;  528          1056  1 !               DC_GROUP
;  529          1057  1 !               DC_SYSTEM
;  530          1058  1 !               DC_NOQUOTA
;  531          1059  1 !               DC_OVR_PROT
;  532          1060  1 !               DC_PROTECTION
;  533          1061  1 !               DC_OVR_OWNUIC
;  534          1062  1 !               DC_OWNER_UIC
;  535          1063  1 !               DC_NOINTERLOCK
;  536          1064  1 !       VOLOCK_STS - lock status block for volume lock
;  537          1065  1 !       VLSETLCK_STS - lock status lock for volume set lock
;  538          1066  1 !       DEVLCK_STS - lock status block for device lock
;  539          1067  1 !
;  540          1068  1 ! Routine value:
;  541          1069  1 !       NONE
;  542          1070  1 !
;  543          1071  1 ! Side effects:
;  544          1072  1 !
;  545          1073  1 !       All process locks are left in their mounted, system owned state
;  546          1074  1 !       if successful.  A full dismount must be done to undo after this.
;  547          1075  1 !       Errors are signalled and the kernel mode handler will undo
;  548          1076  1 !       already converted locks as necessary.
;  549          1077  1 !
;  550          1078  1 !--
;  551          1079  1
;  552          1080  2 BEGIN
;  553          1081  2
;  554          1082  2 BUILTIN
;  555          1083  2         TESTBITCS;
;  556          1084  2
;  557          1085  2 EXTERNAL
;  558          1086  2         MOUNT_OPTIONS   : BITVECTOR,                                    ;
;  559          1087  2         REAL_RVT        : REF BBLOCK,
;  560          1088  2         STORED_CONTEXT  : BITVECTOR,                                    ;
;  561          1089  2         PROTO_VCB       : BBLOCK,
;  562          1090  2         PROTECTION,
;  563          1091  2         OWNER_UIC;
;  564          1092  2
;  565          1093  2 LOCAL
;  566          1094  2         STATUS;
;  567          1095  2
;  568          1096  2 ! Convert the volume lock, if present, to system owned and store the
;  569          1097  2 ! value block.  If this is the first mounter, relevant context
;  570          1098  2 ! in the value block (e.g., volume free space) has already been
;  571          1099  2 ! set up in the value block being stored.
;  572          1100  2 !
;  573          1101  2
;  574          1102  2 IF .VOLOCK_ID NEQ 0
;  575          1103  2 THEN
;  576          1104  3     BEGIN
;  577          1105  3
;  578          1106  3     VOL_CTX [VC_NOTFIRST_MNT] = 1;
;  579          1107  3
;  580        P 1108  3     STATUS = $ENQW (LKMODE = LCK$K_CRMODE,
```

```
;  581        P 1109  3                    LKSB = VOLOCK_STS,
;  582        P 1110  3                    EFN = MOUNT_EFN,
;  583        P 1111  3                    FLAGS = LCK$M_VALBLK + LCK$M_CONVERT + LCK$M_SYNCSTS
;  584          1112  3                        + LCK$M_CVTSYS + LCK$M_NOQUOTA + LCK$M_NOQUEUE);
;  585          1113  3
;  586          1114  3      IF NOT .STATUS
;  587          1115  3      THEN
;  588          1116  4          BEGIN
;  589          1117  4          ERR_EXIT (.STATUS);
;  590          1118  4          RETURN
;  591          1119  3          END;
;  592          1120  3
;  593          1121  4      IF NOT (STATUS = .VOLOCK_STS [0])
;  594          1122  3      THEN
;  595          1123  4          BEGIN
;  596          1124  4          ERR_EXIT (.STATUS);
;  597          1125  4          RETURN
;  598          1126  3          END;
;  599          1127  3
;  600          1128  2      END;
;  601          1129  2
;  602          1130  2  ! If this is a volume set, convert the volume set lock to system owned
;  603          1131  2  ! and store the value block.
;  604          1132  2  !
;  605          1133  2
;  606          1134  2  IF .VLSETLCK_ID NEQ 0
;  607          1135  2  THEN
;  608          1136  3      BEGIN
;  609          1137  3
;  610          1138  3      VLSETLCK_CTX [VC_NOTFIRST_MNT] = 1;
;  611          1139  3
;  612        P 1140  3      STATUS = $ENQW (LKMODE = LCK$K_NLMODE,
;  613        P 1141  3                      EFN = MOUNT_EFN,
;  614        P 1142  3                      LKSB = VLSETLCK_STS,
;  615        P 1143  3                      FLAGS = LCK$M_CONVERT + LCK$M_CVTSYS + LCK$M_SYNCSTS
;  616          1144  3                          + LCK$M_NOQUOTA + LCK$M_NOQUEUE + LCK$M_VALBLK);
;  617          1145  3
;  618          1146  3      IF NOT .STATUS
;  619          1147  3      THEN
;  620          1148  4          BEGIN
;  621          1149  4          ERR_EXIT (.STATUS);
;  622          1150  4          RETURN
;  623          1151  3          END;
;  624          1152  3
;  625          1153  4      IF NOT (STATUS = .VLSETLCK_STS [0])
;  626          1154  3      THEN
;  627          1155  4          BEGIN
;  628          1156  4          ERR_EXIT (.STATUS);
;  629          1157  4          RETURN
;  630          1158  3          END;
;  631          1159  3
;  632          1160  3  ! This is the only case where we are storing a lock ID in the real structure
;  633          1161  3  ! before all lock conversions are complete.  The kernel mode handler knows
;  634          1162  3  ! how to undo this if the device lock conversion fails.
;  635          1163  3  !
;  636          1164  3
;  637          1165  3      REAL_RVT [RVT$L_STRUCLKID] = .VLSETLCK_ID;
```

```
;  638          1166  3
;  639          1167  2      END;
;  640          1168  2
;  641          1169  2  ! If there is no device lock, we are done.
;  642          1170  2  !
;  643          1171  2
;  644          1172  2  IF .DEVLCK_LKID EQL 0
;  645          1173  2  THEN
;  646          1174  3      BEGIN
;  647          1175  3      VOLOCK_ID = 0;
;  648          1176  3      VLSETLCK_ID = 0;
;  649          1177  3      RETURN
;  650          1178  2      END;
;  651          1179  2
;  652          1180  2  IF TESTBITCS (DEV_CTX [DC_NOTFIRST_MNT])
;  653          1181  2  THEN
;  654          1182  3      BEGIN
;  655          1183  3
;  656          1184  3  ! This is the first mounter of this device, so set up appropriate context
;  657          1185  3  ! in the value block.
;  658          1186  3  !
;  659          1187  3
;  660          1188  3      IF .MOUNT_OPTIONS [OPT_FOREIGN]
;  661          1189  3      THEN
;  662          1190  3          DEV_CTX [DC_FOREIGN] = 1;
;  663          1191  3
;  664          1192  3      IF .MOUNT_OPTIONS [OPT_WRITE]
;  665          1193  3      THEN
;  666          1194  3          DEV_CTX [DC_WRITE] = 1;
;  667          1195  3
;  668          1196  3      IF .MOUNT_OPTIONS [OPT_GROUP]
;  669          1197  3      THEN
;  670          1198  3          DEV_CTX [DC_GROUP] = 1;
;  671          1199  3
;  672          1200  3      IF .MOUNT_OPTIONS [OPT_SYSTEM]
;  673          1201  3      THEN
;  674          1202  3          DEV_CTX [DC_SYSTEM] = 1;
;  675          1203  3
;  676          1204  3      IF .MOUNT_OPTIONS [OPT_NOQUOTA]
;  677          1205  3      THEN
;  678          1206  3          DEV_CTX [DC_NOQUOTA] = 1;
;  679          1207  3
;  680          1208  3      IF .MOUNT_OPTIONS [OPT_PROTECTION]
;  681          1209  3      THEN
;  682          1210  4          BEGIN
;  683          1211  4          DEV_CTX [DC_OVR_PROT] = 1;
;  684          1212  4          DEV_CTX [DC_PROTECTION] = .PROTECTION;
;  685          1213  3          END;
;  686          1214  3
;  687          1215  3      IF .MOUNT_OPTIONS [OPT_OWNER_UIC]
;  688          1216  3      THEN
;  689          1217  4          BEGIN
;  690          1218  4          DEV_CTX [DC_OVR_OWNUIC] = 1;
;  691          1219  4          DEV_CTX [DC_OWNER_UIC] = .OWNER_UIC;
;  692          1220  3          END;
;  693          1221  3
;  694          1222  3      IF NOT .STORED_CONTEXT [XQP]
```

```
;  695          1223  3      THEN
;  696          1224  3          DEV_CTX [DC_NOINTERLOCK] = 1;
;  697          1225  3
;  698          1226  2      END;
;  699          1227  2
;  700          1228  2  ! Always store value block.  If this isn't the first mounter, this
;  701          1229  2  ! simply rewrites the value block recovered.  This will clear any
;  702          1230  2  ! value block not valid conditions as a result of node failures
;  703          1231  2  ! in the cluster.
;  704          1232  2  !
;  705          1233  2
;  706        P 1234  2  STATUS = $ENQW (LKMODE = IF NOT .MOUNT_OPTIONS [OPT_NOSHARE]
;  707        P 1235  2                       THEN LCK$K_CRMODE
;  708        P 1236  2                       ELSE LCK$K_EXMODE,
;  709        P 1237  2                  LKSB = DEVLCK_STS,
;  710        P 1238  2                  EFN = MOUNT_EFN,
;  711        P 1239  2                  FLAGS = LCK$M_CONVERT + LCK$M_CVTSYS + LCK$M_VALBLK
;  712          1240  2                        + LCK$M_SYNCSTS + LCK$M_NOQUOTA);
;  713          1241  2
;  714          1242  2  IF NOT .STATUS
;  715          1243  2  THEN
;  716          1244  3      BEGIN
;  717          1245  3      ERR_EXIT (.STATUS);
;  718          1246  3      RETURN
;  719          1247  2      END;
;  720          1248  2
;  721          1249  3  IF (STATUS = .DEVLCK_STS [0])
;  722          1250  2  THEN
;  723          1251  3      BEGIN
;  724          1252  3      DEVLCK_LKID = 0;
;  725          1253  3      VOLOCK_ID = 0;
;  726          1254  3      VLSETLCK_ID = 0;
;  727          1255  3      END
;  728          1256  2  ELSE
;  729          1257  3      BEGIN
;  730          1258  3      ERR_EXIT (.STATUS);
;  731          1259  3      RETURN
;  732          1260  2      END;
;  733          1261  2
;  734          1262  1  END;                    ! of routine store_context


                                                .EXTRN  REAL_RVT, PROTO_VCB

                                    003C 00000  .ENTRY  STORE_CONTEXT, Save R2,R3,R4,R5          ; 0995
                55 00000000G    00   9E 00002  MOVAB   SYS$ENQW, R5                             ;
                54      0000G   CF   9E 00009  MOVAB   MOUNT_OPTIONS, R4                       ;
                53      0000'   CF   9E 0000E  MOVAB   DEV_CTX, R3                             ;
                          14    A3   D5 00013  TSTL    VOLOCK_ID                              ; 1102
                                27   13 00016  BEQL    1$                                     ;
                          18    A3   01   88 00018  BISB2   #1, VOL_CTX                       ; 1106
                                7E   7C 0001C  CLRQ    -(SP)                                  ; 1112
                                7E   7C 0001E  CLRQ    -(SP)                                  ;
                                7E   7C 00020  CLRQ    -(SP)                                  ;
                                7E   D4 00022  CLRL    -(SP)                                  ;
                             7E      6F   8F   9A 00024  MOVZBL  #111, -(SP)                   ;
```

```
                                10  A3  9F 00028          PUSHAB  VOLOCK_STS                              ;
                                    01  DD 0002B          PUSHL   #1                                      ;
                                    1A  DD 0002D          PUSHL   #26                                     ;
                            65      0B  FB 0002F          CALLS   #11, SYS$ENQW                           ;
                            52      50  D0 00032          MOVL    R0, STATUS                              ;
                            2F      52  E9 00035          BLBC    STATUS, 2$                              ; 1114
                            52  10  A3  3C 00038          MOVZWL  VOLOCK_STS, STATUS                      ; 1121
                            28      52  E9 0003C          BLBC    STATUS, 2$                              ;
                                30  A3  D5 0003F  1$:      TSTL    VLSETLCK_ID                             ; 1134
                                    2F  13 00042          BEQL    4$                                      ;
                        34  A3          01  88 00044      BISB2   #1, VLSETLCK_CTX                        ; 1138
                                        7E  7C 00048      CLRQ    -(SP)                                   ; 1144
                                        7E  7C 0004A      CLRQ    -(SP)                                   ;
                                        7E  7C 0004C      CLRQ    -(SP)                                   ;
                                        7E  D4 0004E      CLRL    -(SP)                                   ;
                            7E      6F  8F  9A 00050      MOVZBL  #111, -(SP)                             ;
                                2C  A3  9F 00054          PUSHAB  VLSETLCK_STS                            ;
                            7E      1A  7D 00057          MOVQ    #26, -(SP)                              ;
                            65      0B  FB 0005A          CALLS   #11, SYS$ENQW                           ;
                            52      50  D0 0005D          MOVL    R0, STATUS                              ;
                            04      52  E9 00060          BLBC    STATUS, 2$                              ; 1146
                            52  2C  A3  3C 00063          MOVZWL  VLSETLCK_STS, STATUS                    ; 1153
                            03      52  E8 00067  2$:      BLBS    STATUS, 3$                              ;
                                0095    31 0006A          BRW     17$                                     ;
                    0000G   DF      30  A3  D0 0006D  3$:  MOVL    VLSETLCK_ID, @REAL_RVT                  ; 1165
                                FC  A3  D5 00073  4$:      TSTL    DEVLCK_LRID                             ; 1172
                                    03  12 00076          BNEQ    5$                                      ;
                                0080    31 00078          BRW     16$                                     ;
                4E          63          00  E2 0007B  5$:  BBSS    #0, DEV_CTX, 13$                        ; 1180
                03      01  A4          03  E1 0007F      BBC     #3, MOUNT_OPTIONS+1, 6$                 ; 1188
                            63          02  88 00084      BISB2   #2, DEV_CTX                             ; 1190
                03      01  A4          01  E1 00087  6$:  BBC     #1, MOUNT_OPTIONS+1, 7$                 ; 1192
                            63          10  88 0008C      BISB2   #16, DEV_CTX                            ; 1194
                                        64  95 0008F  7$:  TSTB    MOUNT_OPTIONS                           ; 1196
                                        03  18 00091      BGEQ    8$                                      ;
                            63          04  88 00093      BISB2   #4, DEV_CTX                             ; 1198
                            03      01  A4  E9 00096  8$:  BLBC    MOUNT_OPTIONS+1, 9$                     ; 1200
                            63          08  88 0009A      BISB2   #8, DEV_CTX                             ; 1202
                03      05  A4          02  E1 0009D  9$:  BBC     #2, MOUNT_OPTIONS+5, 10$                ; 1204
                            63          20  88 000A2      BISB2   #32, DEV_CTX                            ; 1206
                0A      02  A4          01  E1 000A5  10$: BBC     #1, MOUNT_OPTIONS+2, 11$                ; 1208
                            63      40  8F  88 000AA      BISB2   #64, DEV_CTX                            ; 1211
                        02  A3  0000G  CF  B0 000AE      MOVW    PROTECTION, DEV_CTX+2                   ; 1212
                0A      02  A4          02  E1 000B4  11$: BBC     #2, MOUNT_OPTIONS+2, 12$                ; 1215
                            63      80  8F  88 000B9      BISB2   #128, DEV_CTX                           ; 1218
                        04  A3  0000G  CF  D0 000BD      MOVL    OWNER_UIC, DEV_CTX+4                    ; 1219
                04  0000G  CF          02  E0 000C3  12$: BBS     #2, STORED_CONTEXT, 13$                 ; 1222
                    01  A3              01  88 000C9      BISB2   #1, DEV_CTX+1                           ; 1224
                                        7E  7C 000CD  13$: CLRQ    -(SP)                                   ; 1240
                                        7E  7C 000CF      CLRQ    -(SP)                                   ;
                                        7E  7C 000D1      CLRQ    -(SP)                                   ;
                                        7E  D4 000D3      CLRL    -(SP)                                   ;
                            7E      6B  8F  9A 000D5      MOVZBL  #107, -(SP)                             ;
                                F8  A3  9F 000D9          PUSHAB  DEVLCK_STS                              ;
                04          64          04  E0 000DC      BBS     #4, MOUNT_OPTIONS, 14$                  ;
                                        01  DD 000E0      PUSHL   #1                                      ;
                                        02  11 000E2      BRB     15$                                     ;
```

```
                                 05  DD 000E4 14$:    PUSHL   #5                                              ;
                                 1A  DD 000E6 15$:    PUSHL   #26                                             ;
                              65 0B  FB 000E8         CALLS   #11, SYS$ENQW                                   ;
                              52 50  D0 000EB         MOVL    R0, STATUS                                      ;
                              11 52  E9 000EE         BLBC    STATUS, 17$                                     ; 1242
                              52 F8  A3 3C 000F1      MOVZWL  DEVLCK_STS, STATUS                              ; 1249
                              0A 52  E9 000F5         BLBC    STATUS, 17$                                     ;
                                 FC  A3 D4 000F8      CLRL    DEVLCK_LKID                                     ; 1252
                                 14  A3 D4 000FB 16$:  CLRL    VOLOCK_ID                                       ; 1253
                                 30  A3 D4 000FE      CLRL    VLSETLCK_ID                                     ; 1254
                                     04 00101         RET                                                     ; 1249
                                 52  DD 00102 17$:    PUSHL   STATUS                                          ; 1258
                  00000000G  00      01  FB 00104     CALLS   #1, LIB$STOP                                    ;
                                     04 0010B         RET                                                     ; 1262

; Routine Size:  268 bytes,    Routine Base:  $CODE$ + 016F
```

```
;  736     1263  1 GLOBAL ROUTINE GET_VOLUME_LOCK_NAME : NOVALUE =
;  737     1264  1
;  738     1265  1 !++
;  739     1266  1 !
;  740     1267  1 ! Functional description:
;  741     1268  1 !
;  742     1269  1 ! This routine generates and stores the resource name used for the
;  743     1270  1 ! volume (allocation) lock in the VCB.
;  744     1271  1 !
;  745     1272  1 ! Input parameters:
;  746     1273  1 !       NONE
;  747     1274  1 !
;  748     1275  1 ! Implicit inputs:
;  749     1276  1 !
;  750     1277  1 !       MOUNT_OPTIONS [OPT_NOSHARE] - set if not a shared mount
;  751     1278  1 !       SCS$GB_NODENAME - unique 8 byte node identifier
;  752     1279  1 !       DEVLCK_UCB - address of UCB (of device being mounted)
;  753     1280  1 !       DEV_CTX [DC_NOTFIRST_MNT] - set if not the first mounter
;  754     1281  1 !       DEV_CTX [DC_WRITE] - set if volume mounted for write access
;  755     1282  1 !       PROTO_VCB [VCB$T_VOLNAME] - volume label
;  756     1283  1 !       BUFFER [SCB$T_VOLOCKNAME] - lock name for already mounted disk
;  757     1284  1 !
;  758     1285  1 ! Output parameters:
;  759     1286  1 !       NONE
;  760     1287  1 !
;  761     1288  1 ! Implicit outputs:
;  762     1289  1 !
;  763     1290  1 !       PROTO_VCB [VCB$T_VOLCKNAM]
;  764     1291  1 !       PROTO_VCB [VCB$V_NOSHARE] - set if non-shared mount
;  765     1292  1 !
;  766     1293  1 ! Routine value:
;  767     1294  1 !       NONE
;  768     1295  1 !
;  769     1296  1 ! Side effects:
;  770     1297  1 !       NONE
;  771     1298  1 !
;  772     1299  1 !--
;  773     1300  1
;  774     1301  2 BEGIN
;  775     1302  2
;  776     1303  2 EXTERNAL
;  777     1304  2         BUFFER                  : BBLOCK,
;  778     1305  2         MOUNT_OPTIONS           : BITVECTOR,
;  779     1306  2         PROTO_VCB               : BBLOCK,
;  780     1307  2         SCS$GB_NODENAME         : ADDRESSING_MODE (GENERAL);
;  781     1308  2
;  782     1309  2 ! If this is a non-shared mount, the resource name is a unique
;  783     1310  2 ! node identifier plus a unique device identifier.
;  784     1311  2 !
;  785     1312  2
;  786     1313  2 IF .MOUNT_OPTIONS [OPT_NOSHARE]
;  787     1314  2 THEN
;  788     1315  3     BEGIN
;  789     1316  3     CH$MOVE (8, SCS$GB_NODENAME, PROTO_VCB [VCB$T_VOLCKNAM]);
;  790     1317  3     (PROTO_VCB [VCB$T_VOLCKNAM] + 8) = .DEVLCK_UCB;
;  791     1318  3     PROTO_VCB [VCB$V_NOSHARE] = 1;
;  792     1319  3     END
```

```
;  793          1320  3
;  794          1321  3 ! For shared mounts, the resource name is the volume label. Because
;  795          1322  3 ! volume labels may change after the volume is mounted, the first
;  796          1323  3 ! mounter will write back the volume label used into the VOLOCKNAME
;  797          1324  3 ! field in the SCB, which is where non-first mounters get it from.
;  798          1325  3 ! Other checks being made guarantee that this name is unique throughout
;  799          1326  3 ! the cluster.
;  800          1327  3 !
;  801          1328  3
;  802          1329  2 ELSE
;  803          1330  2     IF .DEV_CTX [DC_NOTFIRST_MNT] AND .DEV_CTX [DC_WRITE]
;  804          1331  2     AND NOT .MOUNT_OPTIONS [OPT_FOREIGN]
;  805          1332  2     THEN
;  806          1333  2         CH$MOVE (12, BUFFER [SCB$T_VOLOCKNAME], PROTO_VCB [VCB$T_VOLCKNAM])
;  807          1334  2     ELSE
;  808          1335  2         CH$MOVE (12, PROTO_VCB [VCB$T_VOLNAME], PROTO_VCB [VCB$T_VOLCKNAM]);
;  809          1336  2
;  810          1337  1 END;

                                                        .EXTRN  BUFFER, SCS$GB_NODENAME

                                                007C 00000      .ENTRY  GET_VOLUME_LOCK_NAME, Save R2,R3,R4,R5,R6    ; 1263
                     56      0000G   CF   9E 00002      MOVAB   PROTO_VCB+128, R6                                    ;
                13       0000G   CF       04  E1 00007      BBC     #4, MOUNT_OPTIONS, 1$                            ; 1313
                66 00000000G   00         08  28 0000D      MOVC3   #8, SCS$GB_NODENAME, PROTO_VCB+128               ; 1316
                        08   A6   0000'   CF  D0 00015      MOVL    DEVLCK_UCB, PROTO_VCB+136                        ; 1317
                        D3   A6           20  88 0001B      BISB2   #32, PROTO_VCB+83                                ; 1318
                                          04  0001F      RET                                                          ; 1313
                     13      0000'   CF   E9 00020 1$:   BLBC    DEV_CTX, 2$                                      ; 1330
                0D      0000'   CF       04  E1 00025      BBC     #4, DEV_CTX, 2$                                  ;
                07      0000G   CF       03  E0 0002B      BBS     #3, MOUNT_OPTIONS+1, 2$                          ; 1331
                66      0000G   CF       0C  28 00031      MOVC3   #12, BUFFER+34, PROTO_VCB+128                    ; 1333
                                          04  00037      RET                                                          ;
                66        94    A6       0C  28 00038 2$:   MOVC3   #12, PROTO_VCB+20, PROTO_VCB+128              ; 1335
                                          04  0003D      RET                                                          ; 1337

; Routine Size:  62 bytes,    Routine Base:  $CODE$ + 027B
```

```
;  812          1338  1 GLOBAL ROUTINE GET_VOLUME_LOCK =
;  813          1339  1
;  814          1340  1 !++
;  815          1341  1 !
;  816          1342  1 ! Functional description:
;  817          1343  1 !
;  818          1344  1 ! This routine acquires the volume allocation lock in PW mode.
;  819          1345  1 ! This is necessary to allow the value block to be written.
;  820          1346  1 ! If this is a non-shared mount, the system lock will be used
;  821          1347  1 ! as a parent to cause the lock to be mastered locally without
;  822          1348  1 ! any cluster message traffic from the lock manager.
;  823          1349  1 !
;  824          1350  1 ! It also performs a $GETLKIW function on the volume allocation
;  825          1351  1 ! lock to determine the number of locks granted on that resource.
;  826          1352  1 ! This is used later to determine whether a rebuild is necessary
;  827          1353  1 ! on the volume after it is mounted.
;  828          1354  1 !
;  829          1355  1 ! This routine is called in kernel mode.
;  830          1356  1 !
;  831          1357  1 ! Input parameters:
;  832          1358  1 !       NONE
;  833          1359  1 !
;  834          1360  1 ! Implicit inputs:
;  835          1361  1 !
;  836          1362  1 !       PROTO_VCB [VCB$V_NOSHARE] - set if nonshared mount
;  837          1363  1 !       PROTO_VCB [VCB$T_VOLCKNAM] - resource name string
;  838          1364  1 !       EXE$GL_SYSID_LOCK - lock ID of system lock
;  839          1365  1 !
;  840          1366  1 ! Output parameters:
;  841          1367  1 !       NONE
;  842          1368  1 !
;  843          1369  1 ! Implicit outputs:
;  844          1370  1 !
;  845          1371  1 !       VOLOCK_STS - status of ENQ request on volume allocation lock
;  846          1372  1 !       PROTO_VCB [VCB$L_VOLLKID] - lock id of volume allocation lock
;  847          1373  1 !       VOLOCK_CNT - count of granted locks on volume allocation lock
;  848          1374  1 !
;  849          1375  1 ! Routine value:
;  850          1376  1 !
;  851          1377  1 !       success if no errors or VALNOTVALID on volume lock request,
;  852          1378  1 !       else error status from failing service.
;  853          1379  1 !
;  854          1380  1 ! Side effects:
;  855          1381  1 !
;  856          1382  1 !       PW mode lock held on volume allocation lock.
;  857          1383  1 !
;  858          1384  1 !--
;  859          1385  1
;  860          1386  2 BEGIN
;  861          1387  2
;  862          1388  2 EXTERNAL
;  863          1389  2         PROTO_VCB               : BBLOCK,
;  864          1390  2         EXE$GL_SYSID_LOCK       : ADDRESSING_MODE (GENERAL),
;  865          1391  2         MOUNT_OPTIONS           : BITVECTOR,
;  866          1392  2         PHYS_NAME               : VECTOR,               ! Decriptor for physical device
;  867          1393  2         DEVICE_INDEX            : VECTOR;               ! index into PHYS_NAME vector
;  868          1394  2
```

```
;  869          1395  2 LOCAL
;  870          1396  2         LOCKNAME        : VECTOR [70,BYTE],
;  871          1397  2         RESNAM_D        : VECTOR [2] INITIAL (LONG (18), LONG (LOCKNAME)),
;  872          1398  2         PARENT_ID,
;  873          1399  2         STATUS,
;  874          1400  2         K,
;  875          1401  2         SFSBLK          : VECTOR [4, WORD];
;  876          1402  2
;  877          1403  2 MAP
;  878          1404  2         PHYS_NAME       : BBLOCKVECTOR [DEVMAX,8];
;  879          1405  2
;  880          1406  2 !
;  881          1407  2 ! Define descriptor vector offsets.
;  882          1408  2 !
;  883          1409  2 MACRO   LEN     = 0,0,32,0%;
;  884          1410  2 MACRO   ADDR    = 4,0,32,0%;
;  885          1411  2
;  886          1412  2 PARENT_ID = 0;
;  887          1413  2
;  888          1414  2 IF .PROTO_VCB [VCB$V_NOSHARE]
;  889          1415  2 THEN
;  890          1416  2     PARENT_ID = .EXE$GL_SYSID_LOCK;
;  891          1417  2
;  892          1418  2 (LOCKNAME [0])<0,32> = 'F11B';
;  893          1419  2 (LOCKNAME [4])<0,16> = '$v';
;  894          1420  2
;  895          1421  2 DECR K FROM 2 TO 1 DO
;  896          1422  2
;  897          1423  3 BEGIN
;  898          1424  3
;  899          1425  3 !
;  900          1426  3 ! The resource name of the volume lock is derived in two ways:
;  901          1427  3 !
;  902          1428  3 !   1.  Mounted Files-11, use the lock name in the VCB (as set up by
;  903          1429  3 !       the GET_VOLUME_LOCK_NAME routine). Resource name isof fixed
;  904          1430  3 !       length (18 bytes, volume label with trailing blanks).
;  905          1431  3 !
;  906          1432  3 !   2.  Mounted foreign, use the full device name, e.g.
;  907          1433  3 !       F11B$v_allocdevnam. Resouce name is of variable length.
;  908          1434  3 !
;  909          1435  3
;  910          1436  3 IF NOT .MOUNT_OPTIONS [OPT_FOREIGN]
;  911          1437  3 THEN
;  912          1438  3     CH$MOVE (12, PROTO_VCB [VCB$T_VOLCKNAM], LOCKNAME [6])
;  913          1439  3 ELSE
;  914          1440  4     BEGIN
;  915          1441  4     CH$MOVE ( .PHYS_NAME [.DEVICE_INDEX,LEN],          ! Length of device name
;  916          1442  4               .PHYS_NAME [.DEVICE_INDEX,ADDR],         ! Addres of device string
;  917          1443  4               LOCKNAME [6] );                          ! Resource name buffer
;  918          1444  4     RESNAM_D [0] = .PHYS_NAME [.DEVICE_INDEX,LEN]+6;   ! Calculate length of resource name
;  919          1445  3     END;
;  920          1446  3
;  921        P 1447  3 STATUS = $ENQW  (LKMODE = LCK$K_PWMODE,
;  922        P 1448  3                  EFN = MOUNT_EFN,
;  923        P 1449  3                  ACMODE = PSL$C_KERNEL,
;  924        P 1450  3                  LKSB = VOLOCK_STS,
;  925        P 1451  3                  FLAGS = LCK$M_VALBLK + LCK$M_SYSTEM + LCK$M_NOQUOTA
```

```
;  926        P 1452  }                          + LCK$M_SYNCSTS,
;  927        P 1453  }                PARID = .PARENT_ID,
;  928          1454  }                RESNAM = RESNAM_D);
;  929          1455  }
;  930          1456  }  IF NOT .STATUS
;  931          1457  }  THEN
;  932          1458  }      RETURN .STATUS;
;  933          1459  }
;  934          1460  }  STATUS = .VOLOCK_STS [0];
;  935          1461  }
;  936          1462  }  IF NOT .STATUS
;  937          1463  }      AND .STATUS<0,16> NEQ SS$_VALNOTVALID
;  938          1464  }  THEN
;  939          1465  }      RETURN .STATUS;
;  940          1466  }
;  941          1467  }  PROTO_VCB [VCB$L_VOLLKID] = .VOLOCK_ID;
;  942          1468  }
;  943          1469  }  LCKCNT_ITM [4,0,32,0] = VOLOCK_COUNT;
;  944          1470  }
;  945        P 1471  }  STATUS = $GETLKIW (EFN = MOUNT_EFN,
;  946        P 1472  }                      LKIDADR = VOLOCK_ID,
;  947        P 1473  }                      ITMLST = LCKCNT_ITM,
;  948          1474  }                      IOSB = STSBLK);
;  949          1475  }
;  950          1476  }  IF NOT .STATUS
;  951          1477  }  THEN
;  952          1478  }      RETURN .STATUS;
;  953          1479  }
;  954          1480  }  !
;  955          1481  }  ! The device lock value block was read by IOC$SEARCH or routine GET_DEVICE_CONTEXT.
;  956          1482  }  ! We just read the the volume lock value block. The following matrix represents
;  957          1483  }  ! the possible states of these two value blocks:
;  958          1484  }  !
;  959          1485  }  !       DEV_CTX [DC_NOTFIRST_MNT]       VOL_CTX [VC_NOTFIRST_MNT]
;  960          1486  }  !  (a)          0                              0
;  961          1487  }  !  (b)          0                              1
;  962          1488  }  !  (c)          1                              0
;  963          1489  }  !  (d)          1                              1
;  964          1490  }  !
;  965          1491  }  ! Cases (a) and (d) are valid (and therefore not interesting).
;  966          1492  }  !
;  967          1493  }  ! Case  (b) shows that we are the first mounter on this device, yet the
;  968          1494  }  !           volume lock already exists. This implies that another volume
;  969          1495  }  !           with the same label is already mounted. This error condition
;  970          1496  }  !           will be detected later on.
;  971          1497  }  !
;  972          1498  }  ! Case  (c) If the device lock has a count of 1, this shows that
;  973          1499  }  !           when we first read the device context, there was another
;  974          1500  }  !           mounter. However, by the time we read the volume context,
;  975          1501  }  !           this mounter has disappeared. Since MOUNTs and DISMOUNTs
;  976          1502  }  !           are interlocked with the device lock, the mounter couldn't
;  977          1503  }  !           have properly dismounted the volume. The only possibility is
;  978          1504  }  !           that the node that originally mounted this volume had crashed
;  979          1505  }  !           within this window.  In this case, clear the device context
;  980          1506  }  !           block and make us the first mounter. Release the volume lock,
;  981          1507  }  !           derive the volume lock name and try again.
;  982          1508  }  !
```

```
;  983          1509  3 !           Otherwise, this scenario would lead to a "VOLALRMNT" error
;  984          1510  3 !           as in case (b) above.
;  985          1511  3 !
;  986          1512  3 !           Note that we're only doing this for shared mounts.
;  987          1513  3 !
;  988          1514  3
;  989          1515  4 IF  ( .DEV_CTX [DC_NOTFIRST_MNT] )
;  990          1516  4 AND ( NOT .VOL_CTX [VC_NOTFIRST_MNT] )
;  991          1517  4 AND ( NOT .MOUNT_OPTIONS [OPT_NOSHARE] )
;  992          1518  3 THEN
;  993          1519  4     BEGIN
;  994          1520  4     LOCAL
;  995          1521  4         DEVLCK_COUNT,                              ! Device lock count
;  996          1522  4         DEVLCK_ITM      : BBLOCK [12+4] INITIAL
;  997          1523  4                           (WORD (4),
;  998          1524  4                            WORD (LKI$_LCKCOUNT),
;  999          1525  4                            LONG (DEVLCK_COUNT),
; 1000          1526  4                            LONG (0),
; 1001          1527  4                            LONG (0)),
; 1002          1528  4         DEVLCK_IOSB     : VECTOR [4,WORD];
; 1003          1529  4
; 1004        P 1530  4     STATUS = $GETLKIW ( EFN     = MOUNT_EFN,     ! Get number of device locks
; 1005        P 1531  4                         LKIDADR = DEVLCK_LKID,
; 1006        P 1532  4                         ITMLST  = DEVLCK_ITM,
; 1007        P 1533  4                         IOSB    = DEVLCK_IOSB );
; 1008          1534  4
; 1009          1535  5     IF  ( .STATUS )                                ! If $GETLKI succeeded and
; 1010          1536  5     AND ( .DEVLCK_IOSB [0] )                       ! number of device locks eq 1
; 1011          1537  5     AND ( .DEVLCK_COUNT EQL 1 )                    ! then make us the first mounter
; 1012          1538  4     THEN
; 1013          1539  5         BEGIN
; 1014          1540  5         DEV_CTX [DC_FLAGS] = 0;                    ! Clear device lock context
; 1015          1541  5         DEV_CTX [DC_PROTECTION] = 0;
; 1016          1542  5         DEV_CTX [DC_OWNER_UIC] = 0;
; 1017          1543  5         $DEQ ( LKID = .VOLLOCK_ID );               ! Release volume lock
; 1018          1544  5         GET_VOLUME_LOCK_NAME ();                   ! Get the volume lock name (this
; 1019          1545  5         END                                        ! time, as the first mounter)
; 1020          1546  4     ELSE
; 1021          1547  4         EXITLOOP;
; 1022          1548  4     END
; 1023          1549  3 ELSE
; 1024          1550  3     EXITLOOP;                                      ! Otherwise, get out of the loop
; 1025          1551  3
; 1026          1552  2 END;                                               ! End of DECR K loop
; 1027          1553  2
; 1028          1554  2 STATUS = .STSBLK [0]
; 1029          1555  2
; 1030          1556  1 END;                          ! of routine get_volume_lock
; INFO#250                L1:1537
; Referenced LOCAL symbol DEVLCK_COUNT is probably not initialized


                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

                                                0004  00074 P.AAO:  .WORD   4                                                    ;
                                                0205  00076         .WORD   517                                                    ;
```

```
                                00000000  00078          .LONG   0                                                       ;
                                00000000  0007C          .LONG   0                                                       ;
                                00000000  00080          .LONG   0                                                       ;

                                                       .EXTRN  EXE$GL_SYSID_LOCK
                                                       .EXTRN  PHYS_NAME, DEVICE_INDEX
                                                       .EXTRN  SYS$GETLKIW, SYS$DEQ

                                                       .PSECT  $CODE$,NOWRT,2

                                          0FFC 00000   .ENTRY  GET_VOLUME_LOCK, Save R2,R3,R4,R5,R6,R7,R8,-;  1338
                                                               R9,R10,R11                                              ;
                            5B 00000000G  00   9E 00002        MOVAB   SYS$GETLKIW, R11                                ;
                            5A      0000' CF   9E 00009        MOVAB   VOLOCK_ID, R10                                  ;
                            5E        8C  AE   9E 0000E        MOVAB   -116(SP), SP                                    ;
                         24 AE          12   D0 00012        MOVL    #18, RESNAM_D                                   ;  1386
                         28 AE        2C AE   9E 00016        MOVAB   LOCKNAME, RESNAM_D+4                            ;
                                         59   D4 0001B        CLRL    PARENT_ID                                       ;  1412
                   07    0000G   CF         05   E1 0001D        BBC     #5, PROTO_VCB+83, 1$                            ;  1414
                            59 00000000G  00   D0 00023        MOVL    EXE$GL_SYSID_LOCK, PARENT_ID                    ;  1416
                         2C AE 42313146   8F   D0 0002A 1$:     MOVL    #1110520134, LOCKNAME                           ;  1418
                         30 AE     7624   8F   B0 00032        MOVW    #30244, LOCKNAME+4                              ;  1419
                            58          02   D0 00038        MOVL    #2, K                                           ;  1454
                   09       0000G   CF         03   E0 0003B 2$:     BBS     #3, MOUNT_OPTIONS+1, 3$                         ;  1436
                32 AE       0000G   CF         0C   28 00041        MOVC3   #12, PROTO_VCB+128, LOCKNAME+6                  ;  1438
                                            21   11 00048        BRB     4$                                              ;
                            56      0000G   CF   D0 0004A 3$:     MOVL    DEVICE_INDEX, R6                                ;  1441
                                 0000GCF46   7F 0004F        PUSHAQ  PHYS_NAME+4[R6]                                 ;  1442
                            50          9E   D0 00054        MOVL    @(SP)+, R0                                      ;
                                 0000GCF46   7F 00057        PUSHAQ  PHYS_NAME[R6]                                   ;  1443
                32 AE       60          9E   28 0005C        MOVC3   @(SP)+, (R0), LOCKNAME+6                        ;
                                 0000GCF46   7F 00061        PUSHAQ  PHYS_NAME[R6]                                   ;  1444
                24 AE       9E          06   C1 00066        ADDL3   #6, @(SP)+, RESNAM_D                            ;
                                         7E   7C 0006B 4$:     CLRQ    -(SP)                                           ;  1454
                                         7E   7C 0006D        CLRQ    -(SP)                                           ;
                                         7E   D4 0006F        CLRL    -(SP)                                           ;
                                         59   DD 00071        PUSHL   PARENT_ID                                       ;
                                      3C AE   9F 00073        PUSHAB  RESNAM_D                                        ;
                                         39   DD 00076        PUSHL   #57                                             ;
                                      FC AA   9F 00078        PUSHAB  VOLOCK_STS                                      ;
                                         04   DD 0007B        PUSHL   #4                                              ;
                                         1A   DD 0007D        PUSHL   #26                                             ;
                      00000000G  00         0B   FB 0007F        CALLS   #11, SYS$ENQW                                   ;
                                 57          50   D0 00086        MOVL    R0, STATUS                                      ;
                                 2E          57   E9 00089        BLBC    STATUS, 6$                                      ;  1456
                                 57       FC AA   3C 0008C        MOVZWL  VOLOCK_STS, STATUS                              ;  1460
                                 07          57   E8 00090        BLBS    STATUS, 5$                                      ;  1462
                            09F0 8F          57   B1 00093        CMPW    STATUS, #2544                                   ;  1463
                                            7E   12 00098        BNEQ    9$                                              ;
                            0000G   CF         6A   D0 0009A 5$:     MOVL    VOLOCK_ID, PROTO_VCB+124                        ;  1467
                            0000'   CF      14 AA   9E 0009F        MOVAB   VOLOCK_COUNT, LCKCNT_ITM+4                      ;  1469
                                            7E   7C 000A5        CLRQ    -(SP)                                           ;  1474
                                            7E   D4 000A7        CLRL    -(SP)                                           ;
                                         28 AE   9F 000A9        PUSHAB  STSBLK                                          ;
                                      0000' CF   9F 000AC        PUSHAB  LCKCNT_ITM                                      ;
                                            5A   DD 000B0        PUSHL   R10                                             ;
                                            1A   DD 000B2        PUSHL   #26                                             ;
```

```
                                6B              07  FB 000B4          CALLS   #7, SYS$GETLKIW                                  ;
                                57              50  D0 000B7          MOVL    R0, STATUS                                      ;
                                5B              57  E9 000BA 6$:      BLBC    STATUS, 9$                                      ; 1476
                                53        EC    AA  E9 000BD          BLBC    DEV_CTX, 8$                                     ; 1515
                                4F        04    AA  E8 000C1          BLBS    VOL_CTX, 8$                                     ; 1516
                          49          0000G   CF  04  E0 000C5          BBS     #4, MOUNT_OPTIONS, 8$                           ; 1517
                    0C    AE            0000' CF  10  28 000CB          MOVC3   #16, P.AAD, DEVLCK_ITM                          ; 1527
                                        10    AE  6E  9E 000D2          MOVAB   DEVLCK_COUNT, DEVLCK_ITM+4                      ; 1519
                                                  7E  7C 000D6          CLRQ    -(SP)                                           ; 1533
                                                  7E  D4 000D8          CLRL    -(SP)                                           ;
                                        10        AE  9F 000DA          PUSHAB  DEVLCK_IOSB                                     ;
                                        1C        AE  9F 000DD          PUSHAB  DEVLCK_ITM                                      ;
                                        E8        AA  9F 000E0          PUSHAB  DEVLCK_LKID                                     ;
                                                  1A  DD 000E3          PUSHL   #26                                             ;
                                6B                    07  FB 000E5          CALLS   #7, SYS$GETLKIW                         ;
                                57                    50  D0 000E8          MOVL    R0, STATUS                              ;
                                26                    57  E9 000EB          BLBC    STATUS, 8$                              ; 1535
                                22              04    AE  E9 000EE          BLBC    DEVLCK_IOSB, 8$                         ; 1536
                                01                    6E  D1 000F2          CMPL    DEVLCK_COUNT, #1                        ; 1537
                                                      1D  12 000F5          BNEQ    8$                                      ;
                                              EC      AA  7C 000F7          CLRQ    DEV_CTX                                 ; 1540
                                                      7E  7C 000FA          CLRQ    -(SP)                                   ; 1543
                                                      7E  D4 000FC          CLRL    -(SP)                                   ;
                                                      6A  DD 000FE          PUSHL   VOLOCK_ID                               ;
                          00000000G             00      04  FB 00100          CALLS   #4, SYS$DEQ                             ;
                              FEB6              CF      00  FB 00107          CALLS   #0, GET_VOLUME_LOCK_NAME                ; 1544
                                                02      58  F5 0010C          SOBGTR  K, 7$                                   ; 1421
                                                        03  11 0010F          BRB     8$                                      ;
                                                    FF27    31 00111 7$:      BRW     2$                                      ;
                                6B                      57  3C 00114 8$:      MOVZWL  STSBLK, STATUS                          ; 1554
                                50                      57  D0 00118 9$:      MOVL    STATUS, R0                              ;
                                                            04 0011B          RET                                             ; 1556

; Routine Size:  284 bytes,    Routine Base:  $CODE$ + 02B9

; 1031          1557  1
```

```
;  1033     1558  1 GLOBAL ROUTINE GET_VOLSET_LOCK : NOVALUE =
;  1034     1559  1
;  1035     1560  1 !++
;  1036     1561  1 !
;  1037     1562  1 ! Functional description:
;  1038     1563  1 !
;  1039     1564  1 ! This routine generates the resource name used to describe the
;  1040     1565  1 ! volume set name.  This is the same namespace used by the normal
;  1041     1566  1 ! volume allocation locks.  Its primary function is to guarantee
;  1042     1567  1 ! that volume and volume set names are unique throughout the cluster.
;  1043     1568  1 !
;  1044     1569  1 ! This routine is called in kernel mode.
;  1045     1570  1 !
;  1046     1571  1 ! Input parameters:
;  1047     1572  1 !       NONE
;  1048     1573  1 !
;  1049     1574  1 ! Implicit inputs:
;  1050     1575  1 !
;  1051     1576  1 !       HOME_BLOCK [HM2$T_STRUCNAME] - volume set structure name
;  1052     1577  1 !       MOUNT_OPTIONS [OPT_NOSHARE] - set if nonshared mount
;  1053     1578  1 !       SCS$GB_NODENAME - 8 byte unique node identifier
;  1054     1579  1 !       EXE$GL_SYSID_LOCK - lock ID of system (node) lock
;  1055     1580  1 !       REAL_RVT - address of RVT structure
;  1056     1581  1 !       STORED_CONTEXT [XQP] - set for xqp serviced volumes
;  1057     1582  1 !
;  1058     1583  1 ! Output parameters:
;  1059     1584  1 !       NONE
;  1060     1585  1 !
;  1061     1586  1 ! Implicit outputs:
;  1062     1587  1 !
;  1063     1588  1 !       REAL_RVT [RVT$T_VLSLCKNAM] - unique volume set identifier string
;  1064     1589  1 !       VOLSETLCK_STS - status of volume set lock ENQW request
;  1065     1590  1 !       VOLSETLCK_ID - lock ID of volume set lock
;  1066     1591  1 !       VOLSETLCK_CTX - value block of volume set lock
;  1067     1592  1 !
;  1068     1593  1 ! Routine value:
;  1069     1594  1 !       NONE
;  1070     1595  1 !
;  1071     1596  1 ! Side effects:
;  1072     1597  1 !
;  1073     1598  1 !       Error conditions are signalled.
;  1074     1599  1 !       Volume set lock is held in PW mode by this process.
;  1075     1600  1 !
;  1076     1601  1 !--
;  1077     1602  1
;  1078     1603  2 BEGIN
;  1079     1604  2
;  1080     1605  2 EXTERNAL
;  1081     1606  2         HOME_BLOCK              : BBLOCK,
;  1082     1607  2         MOUNT_OPTIONS           : BITVECTOR,
;  1083     1608  2         REAL_RVT                : REF BBLOCK,
;  1084     1609  2         STORED_CONTEXT          : BITVECTOR,
;  1085     1610  2         SCS$GB_NODENAME         : ADDRESSING_MODE (GENERAL),
;  1086     1611  2         EXE$GL_SYSID_LOCK       : ADDRESSING_MODE (GENERAL);
;  1087     1612  2
;  1088     1613  2 LOCAL
;  1089     1614  2         LOCKNAME        : VECTOR [20, BYTE],
```

```
;  1090          1615  2          RESNAM_D        : VECTOR [2] INITIAL (LONG (18), LONG (LOCKNAME)),
;  1091          1616  2          PARENT_ID,
;  1092          1617  2          STATUS;
;  1093          1618  2
;  1094          1619  2
;  1095          1620  2      PARENT_ID = 0;
;  1096          1621  2
;  1097          1622  2      IF .MOUNT_OPTIONS [OPT_NOSHARE]
;  1098          1623  2      THEN
;  1099          1624  2          BEGIN
;  1100          1625  2          CH$MOVE (8, SCS$GB_NODENAME, REAL_RVT [RVT$T_VLSLCKNAM]);
;  1101          1626  2          (REAL_RVT [RVT$T_VLSLCKNAM] + 8) = .REAL_RVT;
;  1102          1627  2          PARENT_ID = .EXE$GL_SYSID_LOCK;
;  1103          1628  2          END
;  1104          1629  2
;  1105          1630  2      ELSE
;  1106          1631  2          CH$MOVE (12, HOME_BLOCK [HM2$T_STRUCNAME], REAL_RVT [RVT$T_VLSLCKNAM]);
;  1107          1632  2
;  1108          1633  2      IF NOT .STORED_CONTEXT [XQP]
;  1109          1634  2      THEN
;  1110          1635  2          RETURN;
;  1111          1636  2
;  1112          1637  2      (LOCKNAME [0])<0,32> = 'F11B';
;  1113          1638  2      (LOCKNAME [4])<0,16> = '$v';
;  1114          1639  2
;  1115          1640  2      CH$MOVE (12, REAL_RVT [RVT$T_VLSLCKNAM], LOCKNAME [6]);
;  1116          1641  2
;  1117          1642  2      ! Take out a lock on the volume set name.
;  1118          1643  2      !
;  1119          1644  2
;  1120        P 1645  2      STATUS = $ENQW (LKMODE = LCK$K_PWMODE,
;  1121        P 1646  2                      EFN = MOUNT_EFN,
;  1122        P 1647  2                      ACMODE = PSL$C_KERNEL,
;  1123        P 1648  2                      RESNAM = RESNAM_D,
;  1124        P 1649  2                      PARID = .PARENT_ID,
;  1125        P 1650  2                      LKSB = VLSETLCK_STS,
;  1126        P 1651  2                      FLAGS = LCK$M_SYSTEM + LCK$M_NOQUOTA + LCK$M_SYNCSTS
;  1127          1652  2                              + LCK$M_VALBLK);
;  1128          1653  2
;  1129          1654  2      IF NOT .STATUS
;  1130          1655  2      THEN
;  1131          1656  3          BEGIN
;  1132          1657  3          ERR_EXIT (.STATUS);
;  1133          1658  3          RETURN
;  1134          1659  2          END;
;  1135          1660  2
;  1136          1661  3      IF NOT (STATUS = .VLSETLCK_STS [0])
;  1137          1662  2              AND .VLSETLCK_STS [0] NEQ SS$_VALNOTVALID
;  1138          1663  2      THEN
;  1139          1664  3          BEGIN
;  1140          1665  3          ERR_EXIT (.STATUS);
;  1141          1666  3          RETURN
;  1142          1667  2          END;
;  1143          1668  2
;  1144          1669  1 END;
```

```
                                                        .EXTRN  HOME_BLOCK

                                                 01FC 00000     .ENTRY  GET_VOLSET_LOCK, Save R2,R3,R4,R5,R6,R7,R8 ; 1558
                                  58      0000'  CF   9E 00002  MOVAB   VLSETLCK_STS, R8
                                  5E             18   C2 00007  SUBL2   #24, SP
                                                 12   DD 0000A  PUSHL   #18                                        ; 1603
                              04  AE        08    AE   9E 0000C  MOVAB   LOCKNAME, RESNAM_D+4
                                                 57   D4 00011  CLRL    PARENT_ID                                 ; 1620
                                  56      0000G  CF   D0 00013  MOVL    REAL_RVT, R6                              ; 1625
                          16      0000G   CF            04   E1 00018  BBC     #4, MOUNT_OPTIONS, 1$                     ; 1622
                      18  A6  00000000G   00            08   28 0001E  MOVC3   #8, SCS$GB_NODENAME, 24(R6)          ; 1625
                              20          A6            56   D0 00027  MOVL    R6, 32(R6)                          ; 1626
                                  57  00000000G         00   D0 0002B  MOVL    EXE$GL_SYSID_LOCK, PARENT_ID        ; 1627
                                                       C7   11 00032  BRB     2$                                  ; 1622
                      18  A6      0000G   CF            0C   28 00034  1$:     MOVC3   #12, HOME_BLOCK+460, 24(R6) ; 1631
                          4A      0000G   CF            02   E1 0003B  2$:     BBC     #2, STORED_CONTEXT, 4$      ; 1633
                                  08  AE  42313146      8F   D0 00041  MOVL    #1110520134, LOCKNAME               ; 1637
                                  0C  AE      7624      8F   B0 00049  MOVW    #30244, LOCKNAME+4                  ; 1638
                  0E  AE          18  A6                0C   28 0004F  MOVC3   #12, 24(R6), LOCKNAME+6            ; 1640
                                                       7E   7C 00055  CLRQ    -(SP)                               ; 1652
                                                       7E   7C 00057  CLRQ    -(SP)
                                                       7E   D4 00059  CLRL    -(SP)
                                                       57   DD 0005B  PUSHL   PARENT_ID
                                              18       AE   9F 0005D  PUSHAB  RESNAM_D
                                                       39   DD 00060  PUSHL   #57
                                                       58   DD 00062  PUSHL   R8
                                                       04   DD 00064  PUSHL   #4
                                                       1A   DD 00066  PUSHL   #26
                                  00000000G   00       0B   FB 00068  CALLS   #11, SYS$ENQW
                                              52       50   D0 0006F  MOVL    R0, STATUS
                                              0D       52   E9 00072  BLBC    STATUS, 3$                          ; 1654
                                              52       68   3C 00075  MOVZWL  VLSETLCK_STS, STATUS               ; 1661
                                              10       52   E8 00078  BLBS    STATUS, 4$                          ; 1662
                                      09F0    8F       68   B1 0007B  CMPW    VLSETLCK_STS, #2544
                                                       09   13 00080  BEQL    4$                                  ; 1665
                                              52       DD   00082  3$:     PUSHL   STATUS
                                  00000000G   00       01   FB 00084  CALLS   #1, LIB$STOP                       ; 1669
                                                       04   0008B  4$:     RET

; Routine Size:  140 bytes,    Routine Base:  $CODE$ + 03D5

; 1145          1670  1
```

```
;  1147          1671  1 ROUTINE KERN_LCK_CLNUP : NOVALUE =
;  1148          1672  1
;  1149          1673  1 !++
;  1150          1674  1 !
;  1151          1675  1 ! Functional description:
;  1152          1676  1 !
;  1153          1677  1 ! This routine is called in kernel mode to back off partial changes
;  1154          1678  1 ! to the locks that mount manipulates.
;  1155          1679  1 ! It backs off locks already converted when an error occurs.
;  1156          1680  1 !
;  1157          1681  1 ! Input parameters:
;  1158          1682  1 !       NONE
;  1159          1683  1 !
;  1160          1684  1 ! Implicit inputs:
;  1161          1685  1 !
;  1162          1686  1 !       VOLOCK_ID - nonzero if the volume lock is to be dequeued.
;  1163          1687  1 !       VLSETLCK_ID - nonzero if the volume set lock is to be dequeued.
;  1164          1688  1 !
;  1165          1689  1 ! Output parameters:
;  1166          1690  1 !       NONE
;  1167          1691  1 !
;  1168          1692  1 ! Implicit outputs:
;  1169          1693  1 !       NONE
;  1170          1694  1 !
;  1171          1695  1 ! Routine value:
;  1172          1696  1 !       NONE
;  1173          1697  1 !
;  1174          1698  1 ! Side effects:
;  1175          1699  1 !
;  1176          1700  1 !       Volume and volume set locks acquired by the MOUNT system service
;  1177          1701  1 !       so far are dequeued (they did not exist previously).
;  1178          1702  1 !
;  1179          1703  1 !--
;  1180          1704  1
;  1181          1705  2 BEGIN
;  1182          1706  2
;  1183          1707  2 IF .VOLOCK_ID NEQ 0
;  1184          1708  2 THEN
;  1185          1709  2     $DEQ (LKID = .VOLOCK_ID);
;  1186          1710  2
;  1187          1711  2 IF .VLSETLCK_ID NEQ 0
;  1188          1712  2 THEN
;  1189          1713  2     $DEQ (LKID = .VLSETLCK_ID);
;  1190          1714  2
;  1191          1715  1 END;


                                0004 00000 KERN_LCK_CLNUP:
                                           .WORD   Save R2                                          ; 1671
52 00000000G   00    9E 00002              MOVAB   SYS$DEQ, R2                                      ;
50      0000'  CF    D0 00009              MOVL    VOLOCK_ID, R0                                    ; 1707
               09    13 0000E              BEQL    1$                                               ;
               7E    7C 00010              CLRQ    -(SP)                                            ; 1709
               7E    D4 00012              CLRL    -(SP)                                            ;
```

```
                        50  DD 00014          PUSHL   R0                                      ;
                    62  04  FB 00016          CALLS   #4, SYS$DEQ                             ;
                50  0000'  CF  D0 00019 1$:    MOVL    VLSETLCK_ID, R0                         ; 1711
                        09  13 0001E          BEQL    2$                                      ;
                        7E  7C 00020          CLRQ    -(SP)                                   ; 1713
                        7E  D4 00022          CLRL    -(SP)                                   ;
                        50  DD 00024          PUSHL   R0                                      ;
                    62  04  FB 00026          CALLS   #4, SYS$DEQ                             ;
                            04 00029 2$:      RET                                             ; 1715

; Routine Size:  42 bytes,    Routine Base:  $CODE$ + 0461
```

```
;  1193          1716  1 GLOBAL ROUTINE LOCK_CLEANUP : NOVALUE =
;  1194          1717  1
;  1195          1718  1 !++
;  1196          1719  1 !
;  1197          1720  1 ! Functional description:
;  1198          1721  1 !
;  1199          1722  1 ! This routine is called from the MOUNT_HANDLER in MOUDK2 when
;  1200          1723  1 ! errors occur.  If any locks have been acquired, it calls a
;  1201          1724  1 ! kernel mode routine to dequeue or convert them as appropriate.
;  1202          1725  1 !
;  1203          1726  1 ! Implicit inputs:
;  1204          1727  1 !
;  1205          1728  1 !       VOLOCK_ID - nonzero if volume lock acquired
;  1206          1729  1 !       VLSETLCK_ID - nonzero if volume set lock acquired
;  1207          1730  1 !
;  1208          1731  1 !--
;  1209          1732  1
;  1210          1733  2 BEGIN
;  1211          1734  2
;  1212          1735  2 IF .VOLOCK_ID NEQ 0
;  1213          1736  2    OR .VLSETLCK_ID NEQ 0
;  1214          1737  2 THEN
;  1215          1738  2     KERNEL_CALL (KERN_LCK_CLNUP);
;  1216          1739  2
;  1217          1740  1 END;


                                                        .EXTRN  SYS$CMKRNL

                                    0000 00000          .ENTRY  LOCK_CLEANUP, Save nothing                      ; 1716
                              0000' CF   D5 00002       TSTL    VOLOCK_ID                                       ; 1735
                                    06   12 00006       BNEQ    1$                                              ;
                              0000' CF   D5 00008       TSTL    VLSETLCK_ID                                     ; 1736
                                    0E   13 0000C       BEQL    2$                                              ;
                                    7E   D4 0000E 1$:   CLRL    -(SP)                                           ; 1738
                                    5E   DD 00010       PUSHL   SP                                              ;
                                C1  AF   9F 00012       PUSHAB  KERN_LCK_CLNUP                                  ;
                    00000000G   9F      03 FB 00015     CALLS   #3, @#SYS$CMKRNL                                ;
                                        04 0001C 2$:    RET                                                     ; 1740

; Routine Size:  29 bytes,    Routine Base:  $CODE$ + 048B

;  1218          1741  1
;  1219          1742  1 END
;  1220          1743  0 ELUDOM


                                                        .EXTRN  LIB$STOP

;                                PSECT SUMMARY
;
;       Name                            Bytes                           Attributes
;
```

```
;  $OWN$          16  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $GLOBAL$       80  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$       1192  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $PLIT$        132  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)



;                              Library Statistics
;
;                                      -------- Symbols --------      Pages       Processing
;       File                           Total    Loaded   Percent      Mapped      Time
;
;  _$255$DUA28:[SYSLIB]LIB.L32;1        18619        43         0      1000          00:01.9



; Information:  1
; Warnings:     0
; Errors:       0



;                                    COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:CLUSTRMNT/OBJ=OBJ$:CLUSTRMNT MSRC$:CLUSTRMNT/UPDATE=(ENH$:CLUSTRMNT)

; Size:         1192 code + 228 data bytes
; Run Time:       00:32.3
; Elapsed Time:   01:03.6
; Lines/CPU Min:   3241
; Lexemes/CPU-Min: 26321
; Memory Used:  180 pages
; Compilation Complete
```

CHKHM2
LIS

CHNUCB
LIS

GETUIC
LIS

ERASE
LIS

LEFTONE
LIS

MOUDK2
LIS

CHKHM1
LIS

CHKSM2
LIS

CLUSTRMNT
LIS

INIFC2
LIS

MOUDK1
LIS

MAKRVT
LIS

MAKLOG
LIS